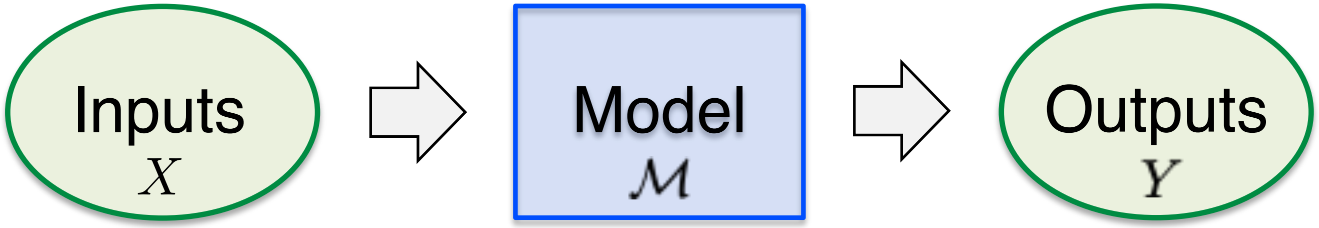
Machine Learning for Uncertainty Quantification: Trusting the Black Box

Team: Jim Warner, Geoffrey Bomarito, Patrick Leser, William Leser (+ too many collaborators to list)
NASA Langley Research Center

Quantification of Uncertainty Across Disciplines Seminar Series

January 25th, 2022

Motivation: Modeling & Simulation

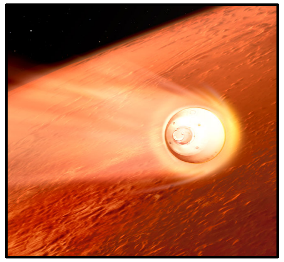


Motivation: Modeling & Simulation

Inputs X → Model $\mathcal{M}$ → Outputs Y

Trajectory Simulation

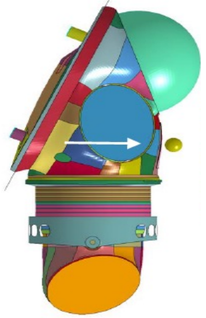
- Atm. conds.
- Mass props



Landing location

Spacesuit Impact Model

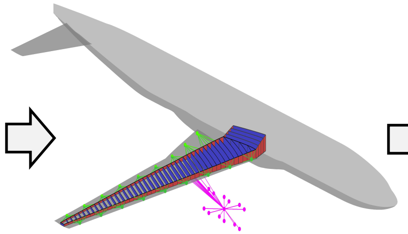
- Impact velocity
- Material strength



Max contact force

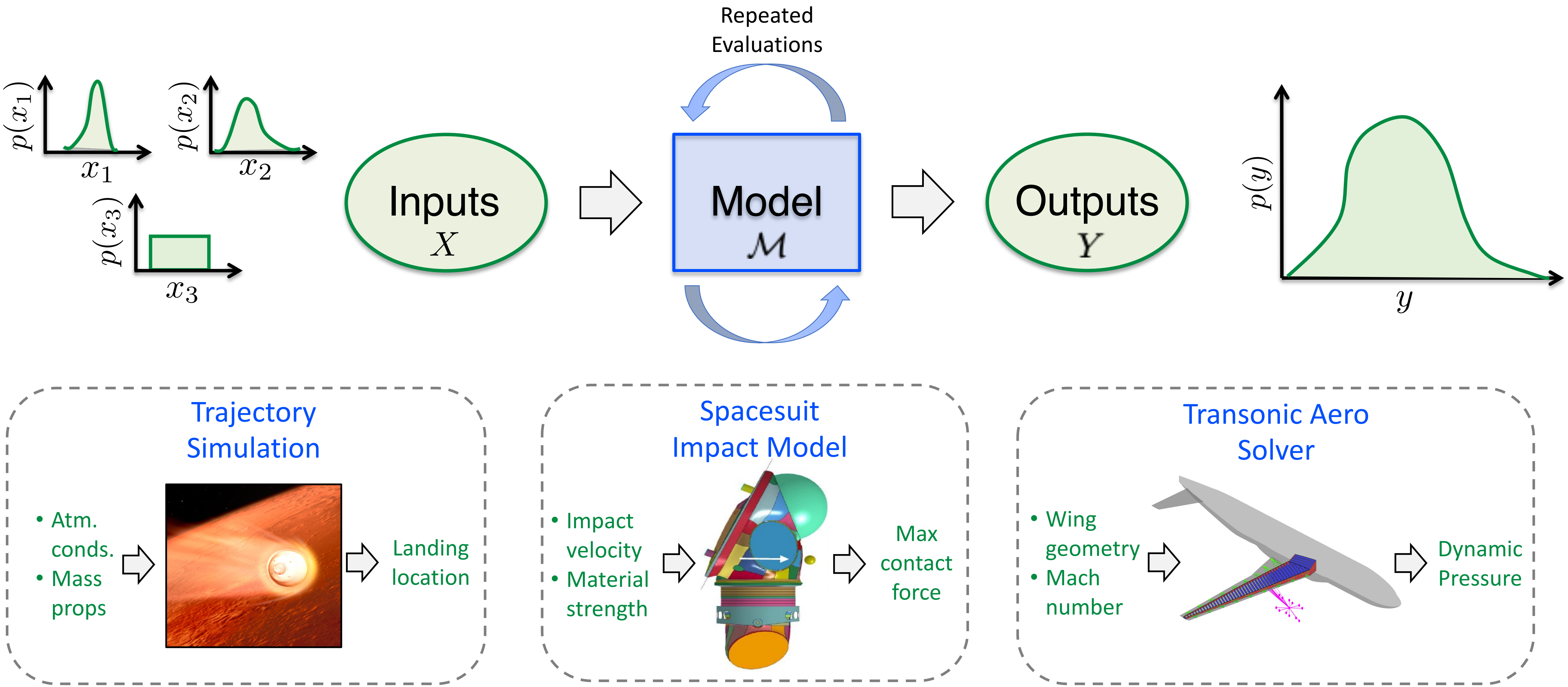
Transonic Aero Solver

- Wing geometry
- Mach number

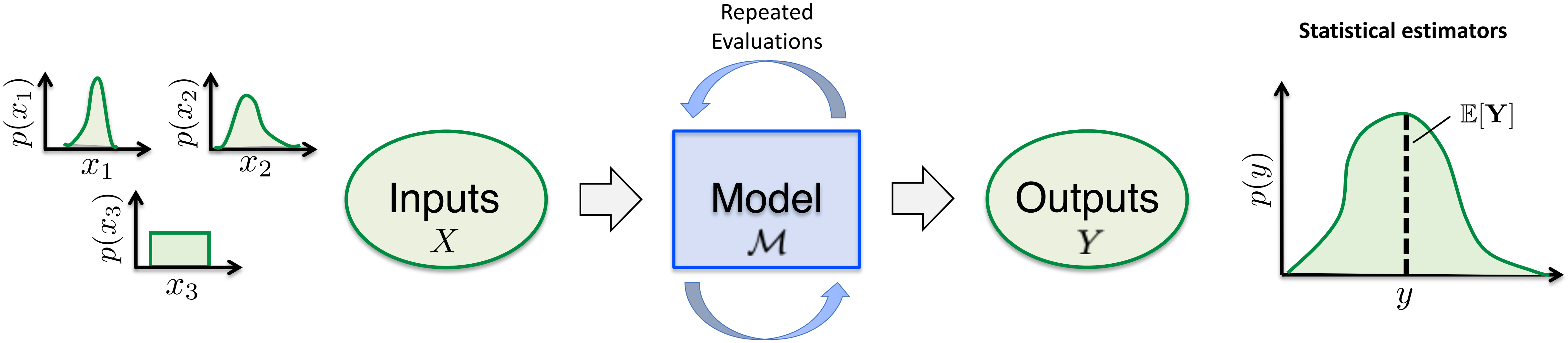


Dynamic Pressure

UQ for Modeling & Simulation

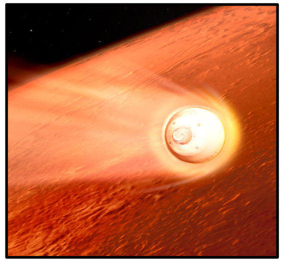


UQ for Modeling & Simulation



Trajectory Simulation

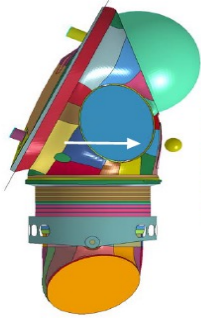
- Atm. conds.
- Mass props



Landing location

Spacesuit Impact Model

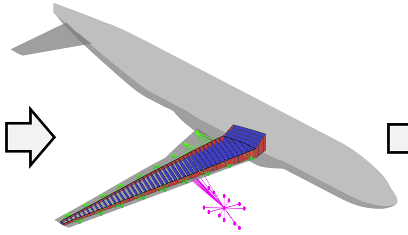
- Impact velocity
- Material strength



Max contact force

Transonic Aero Solver

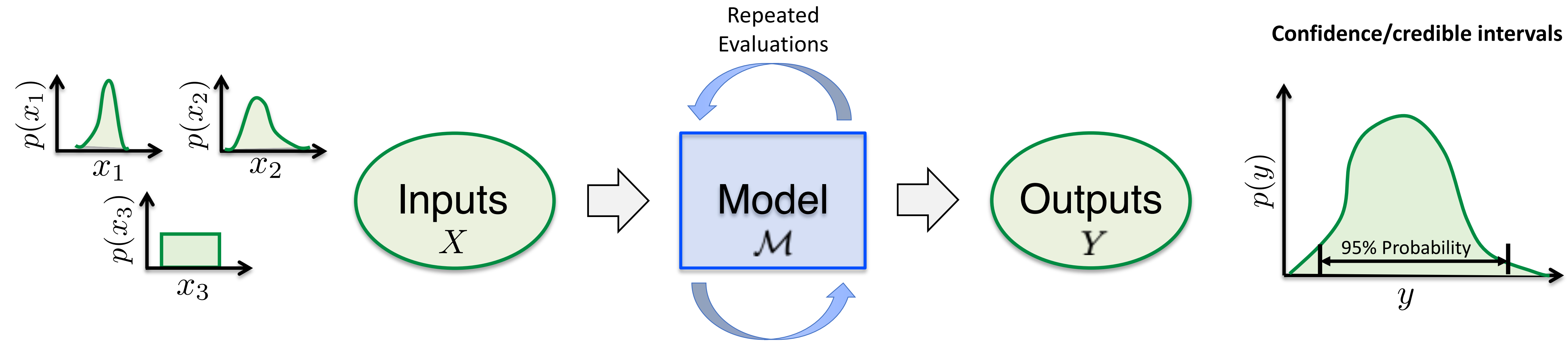
- Wing geometry
- Mach number



Dynamic Pressure

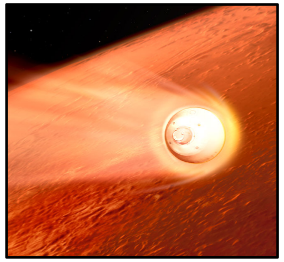
➢ **Expected landing location**

UQ for Modeling & Simulation



Trajectory Simulation

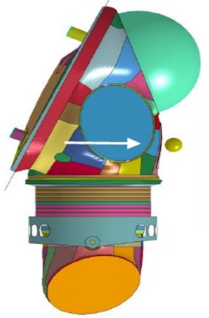
- Atm. conds.
- Mass props



Landing location

Spacesuit Impact Model

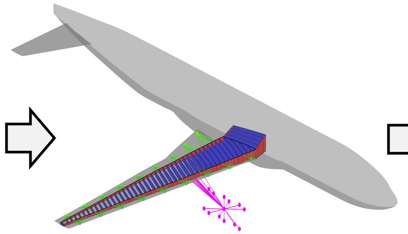
- Impact velocity
- Material strength



Max contact force

Transonic Aero Solver

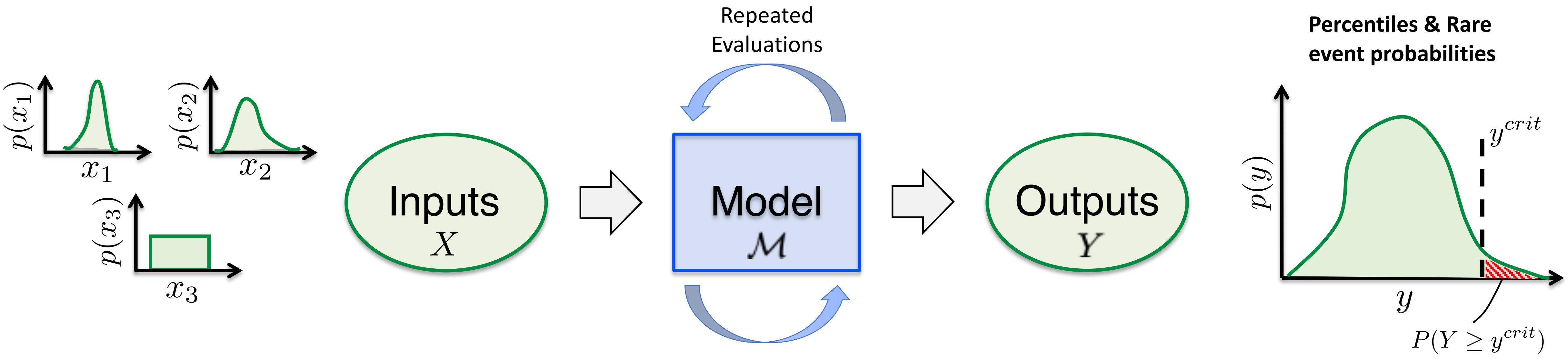
- Wing geometry
- Mach number



Dynamic Pressure

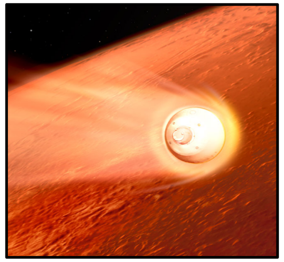
- ➢ Expected landing location
- ➢ **95% confidence landing ellipse**

UQ for Modeling & Simulation



Trajectory Simulation

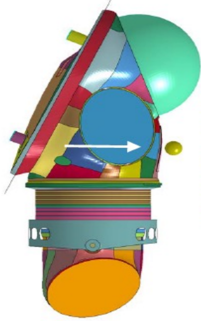
- Atm. conds.
- Mass props



Landing location

Spacesuit Impact Model

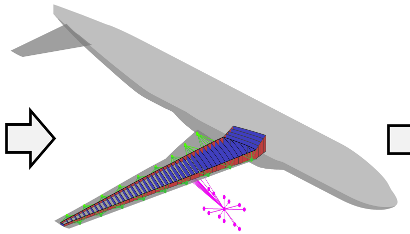
- Impact velocity
- Material strength



Max contact force

Transonic Aero Solver

- Wing geometry
- Mach number



Dynamic Pressure

- ➢ Expected landing location
- ➢ 95% confidence landing ellipse

- ➢ **Probability of failure (force > threshold)**

- ➢ **Probability of flutter**

UQ for Modeling & Simulation

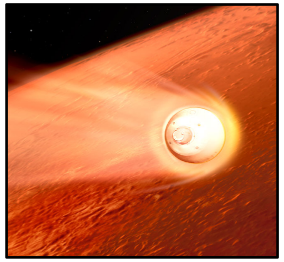
UQ with high-fidelity, physics-based models can be computationally intractable

Number of Model Evaluations O(10^3-10^6)

Inputs X ⇨ Model $\mathcal{M}$ ⇨ Outputs Y

Trajectory Simulation

- Atm. conds.
- Mass props

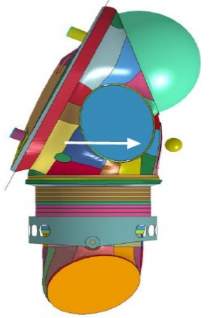


Landing location

Computation Time: O(mins-hours)

Spacesuit Impact Model

- Impact velocity
- Material strength

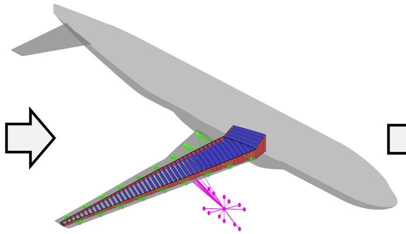


Max contact force

Computation Time: O(hours-days)

Transonic Aero Solver

- Wing geometry
- Mach number

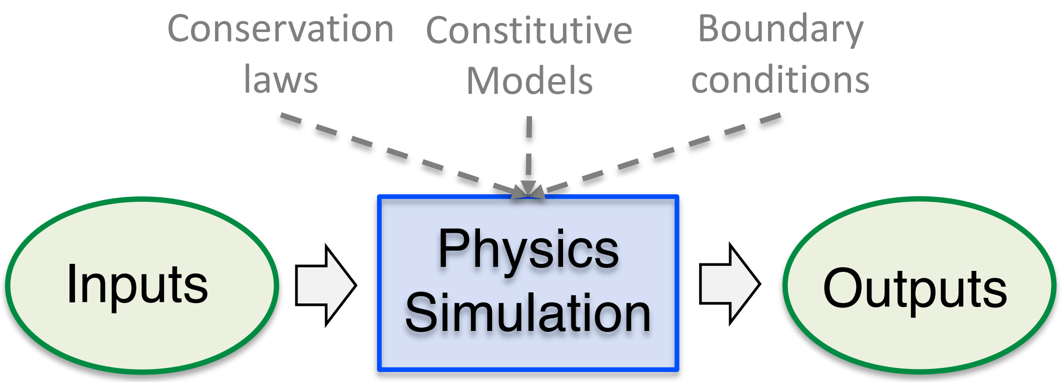


Dynamic Pressure

Computation Time: O(mins-hours)

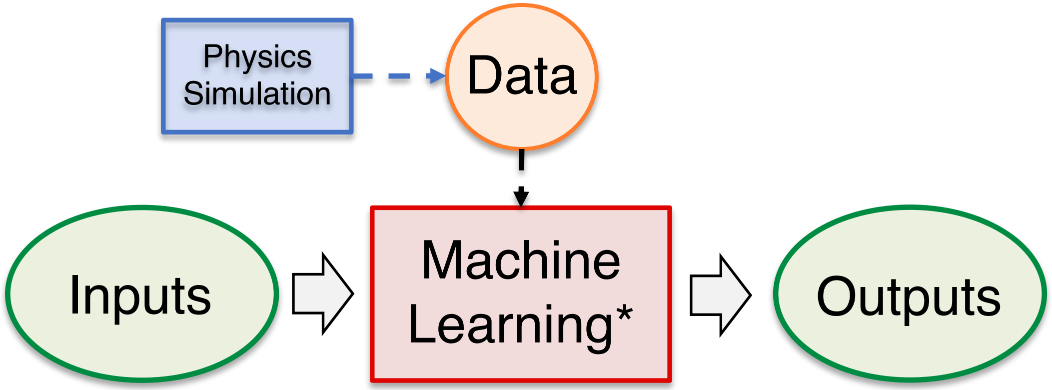
Modeling for UQ

Physics-Based Modeling



Interpretable predictions
Theoretically rigorous
Historically-proven legacy codes
Computationally intensive
Slow development & deployment

Data-Driven Modeling



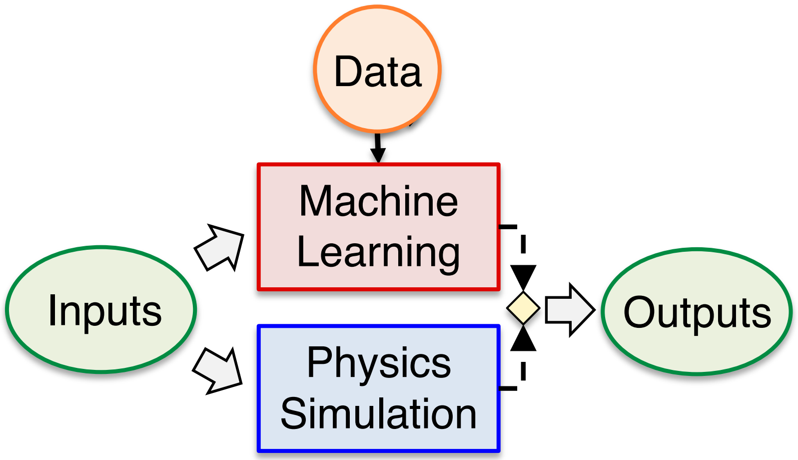
Efficient predictions
Fast deployment
Widely accessible libraries
Lack interpretability
Ineffective for extrapolation/ sparse datasets

*Surrogate model, response surface, reduced-order model, kriging, metamodel, …….

The Best of Both Modeling Worlds

- Exploiting the synergy between physics-based & data-driven modeling

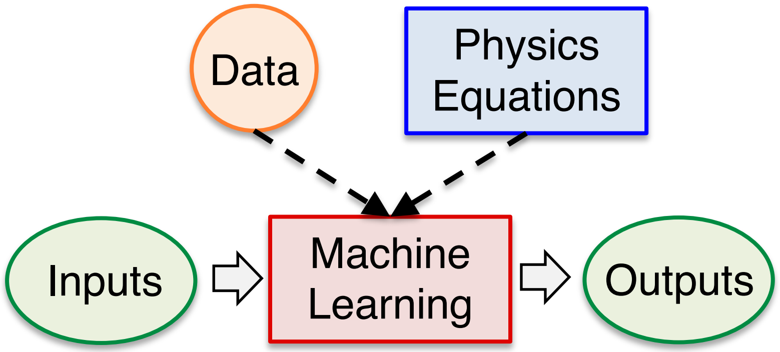
Multi-model Monte Carlo



➢ ***Fuse*** predictions from ML and physics-based models

1. Multilevel Monte Carlo. Giles et al. 2015
2. Multifidelity Monte Carlo. Peherstorfer et al. 2016
3. Approximate Control Variates. Gorodetsky et al. 2019

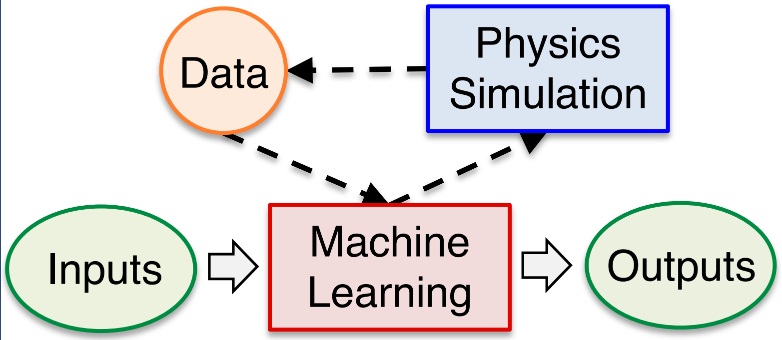
Physics-informed deep learning



➢ ***Integrate*** physical laws into the training of ML models

4. Physics-informed neural networks. Raissi et al. 2019.
5. Physics-informed generative adversarial networks Yang et. al. 2018

Active Learning



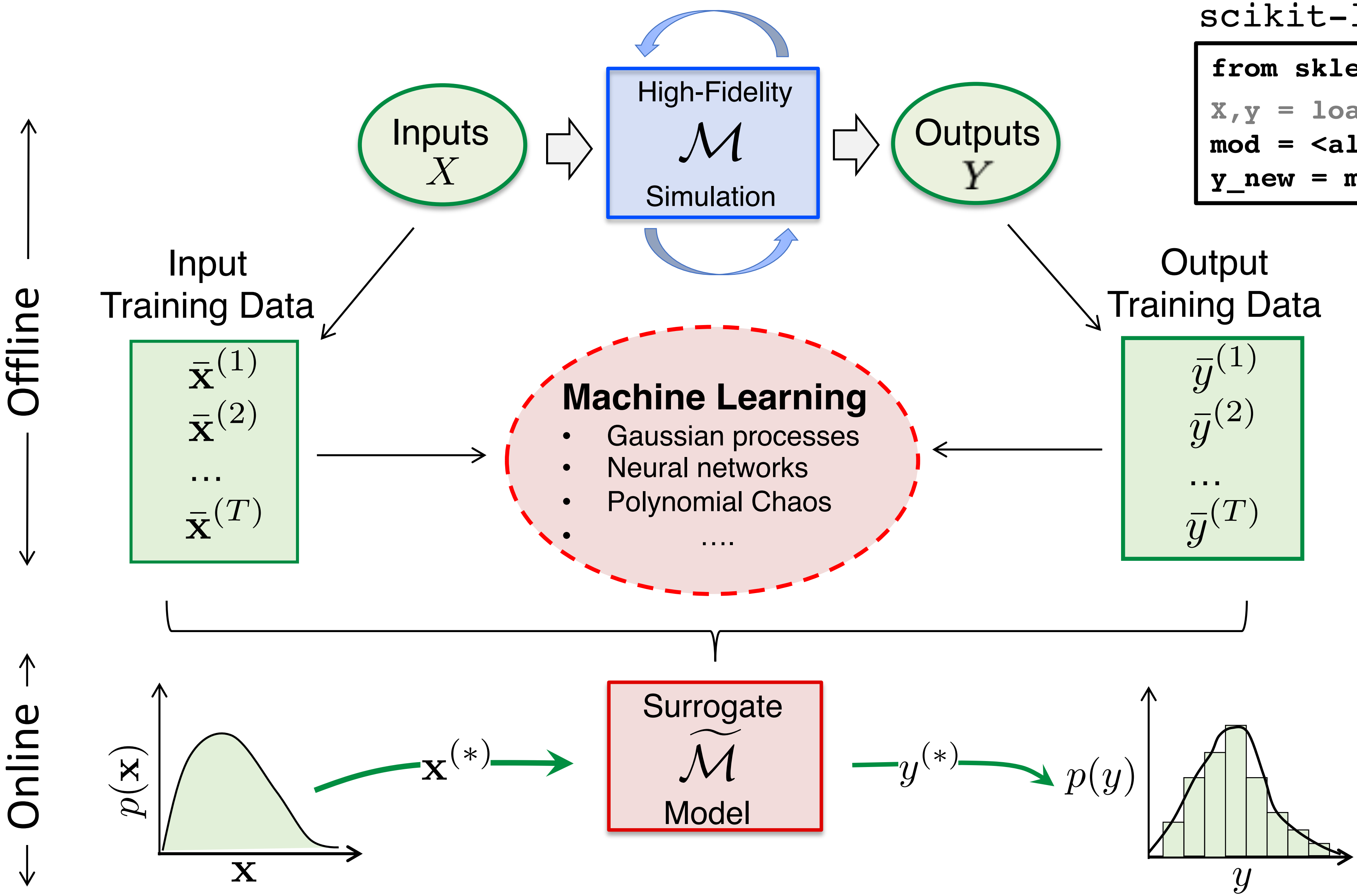
➢ ***Guide*** physics-based training data generation using ML Model

6. Active learning in practice. Settles 2011.
7. Active learning for reliability analysis. Bichon et al. 2008.

Outline

- Surrogate modeling for UQ
- Combining physics-based modeling & machine learning for UQ
  - Multi-model Monte Carlo simulation
    - ➢ Application: Trajectory simulation for EDL
  - Physics-informed generative adversarial networks (PI-GANs)
    - ➢ Application: Material identification
  - Active learning with Gaussian Process models
    - ➢ Application: Reliability analysis
- Summary

Surrogate Modeling



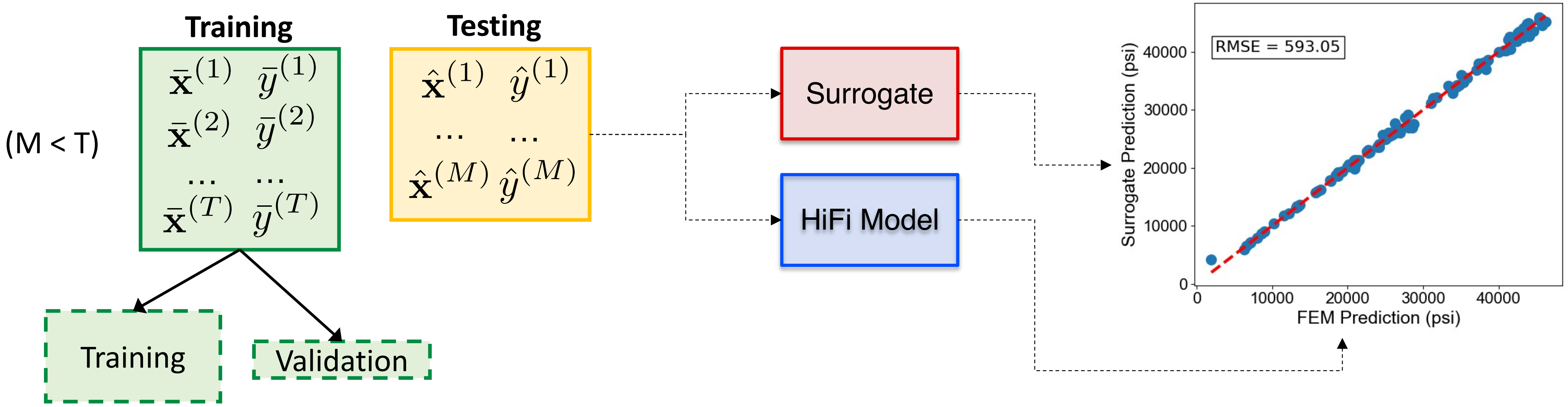
`scikit-learn` with Python:

```
from sklearn import <alg>
X,y = load_data()
mod = <alg>.fit(X, y)
y_new = mod.predict(X_new)
```

Orders of magnitude computational speedup possible

Surrogate Model Validation

- **Always** create a **separate** dataset for testing that is **not** used for training
- A third *validation* dataset is often used during training to tune machine learning *hyperparameters* (e.g., # layers, nodes in neural network)
- If surrogate model error is non-negligible, it can be factored into total uncertainty when making probabilistic predictions

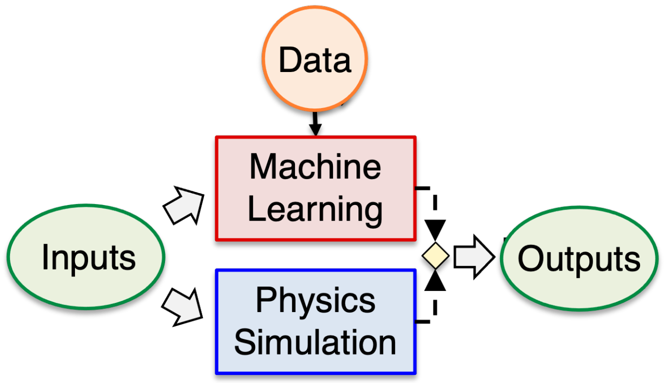


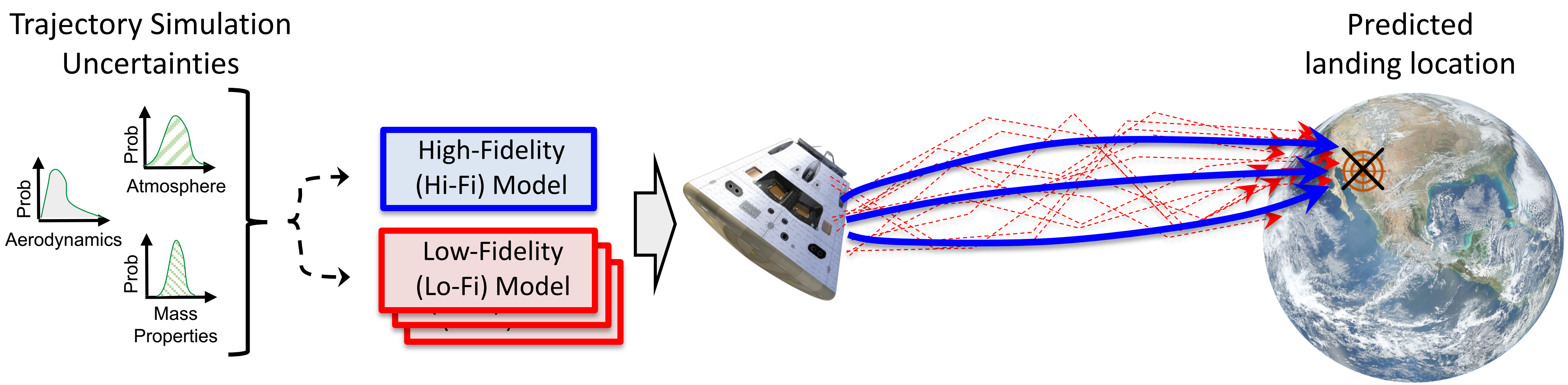
Surrogate Modeling Pitfalls & Challenges

1) Not validating the effectiveness of surrogate prior to utilization
   - ***Or validating the model using the training data!***
2) Not accounting for (or correcting) surrogate errors when they are non-negligible
   - Surrogate models yield **biased** estimators
3) Using surrogate to extrapolate outside the range of training data
4) Training surrogate models from sparse training data
   - Common when high-fidelity model is expensive to run

Multi-Model Monte Carlo for Trajectory Simulation

- **Motivation**: high-precision guidance software for entry vehicles
- **Challenge**: uncertainty propagation with expensive trajectory simulation models
- **Approach**: Use multi-model Monte Carlo to estimate trajectory state *statistics* (e.g., expected landing location)
- **Collaborators**: Anthony Williams (NASA), Som Dutta (NASA), Justin Green (NASA), Luke Morrill (NASA), Sam Nieomoeller (UCLA)





Hi-Fi vs. Lo-Fi Monte Carlo (MC) Estimators

1) Draw N random inputs   2) Evaluate model   3) Form estimator

Hi-Fi MC:

$\{X^{(i)}\}$ → $i = 1, \ldots, N$ **Hi-Fi Model** $Y^{(i)} = \mathcal{M}(X^{(i)})$ → $\{Y^{(i)}\}$

$$\mathbb{E}[Y] \approx \hat{Y} = \frac{1}{N}\sum_{i=1}^{N} Y^{(i)}$$

Unbiased: $\mathbb{E}[\hat{Y}] = \mathbb{E}[Y]$

Solution Time: $N \times C$

$N \sim O(10^4)$

Model cost

Lo-Fi MC:

$\{X^{(i)}\}$ → **ML Surrogate** $\widetilde{Y}^{(i)} = \widetilde{\mathcal{M}}(X^{(i)})$ → $\{\widetilde{Y}^{(i)}\}$

$$\mathbb{E}[Y] \approx \hat{\widetilde{Y}} = \frac{1}{N}\sum_{i=1}^{N} \widetilde{Y}^{(i)}$$

Solution Time: $N \times \widetilde{C}$

Surrogate cost

Biased: $\mathbb{E}[\hat{\widetilde{Y}}] \neq \mathbb{E}[Y]$

Surrogate cost << Model cost

Mean Squared Error = bias + variance

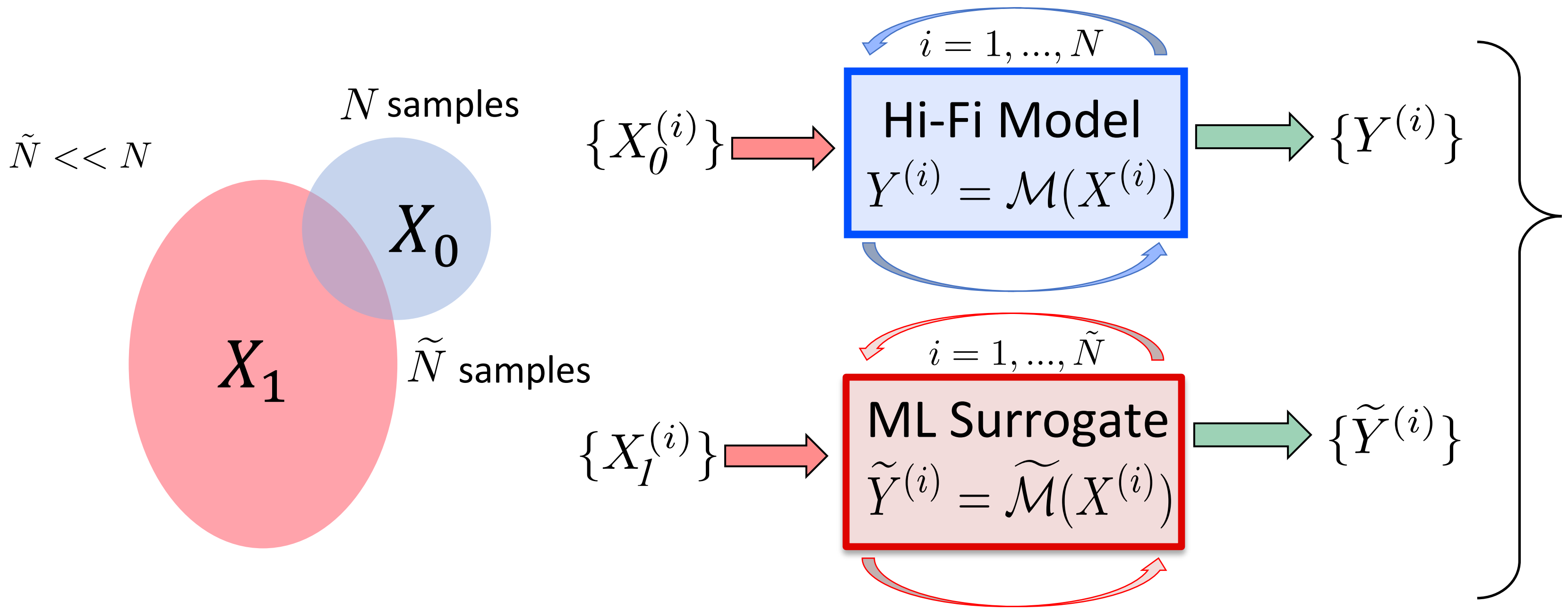
bias → irreducible error!

Multi-Model MC (1/2)

1) Draw & allocate random inputs

2) Evaluate models

3) Form estimator



$$\mathbb{E}[Y] \approx f\left(\{Y^i\}, \{\widetilde{Y}^i\}, N, \widetilde{N}\right)$$

Constructed to be **Unbiased** and **Fast!**

➢ **Main idea**: estimate with a surrogate and correct with high-fidelity model

Ex: *Multi-level MC (MLMC)**

$$\hat{Y}_{\text{MLMC}} = \mathbb{E}[\widetilde{Y}] + \mathbb{E}[Y - \widetilde{Y}]$$

$$\approx \frac{1}{\widetilde{N}} \sum_{i=1}^{\widetilde{N}} \widetilde{Y}^i + \frac{1}{N} \sum_{i=1}^{N} (Y^i - \widetilde{Y}^i)$$

Unbiased

$\mathbb{E}[\hat{Y}_{\text{MLMC}}] = E[Y]$

Solution Time:

$NC + (N + \widetilde{N})\widetilde{C}$

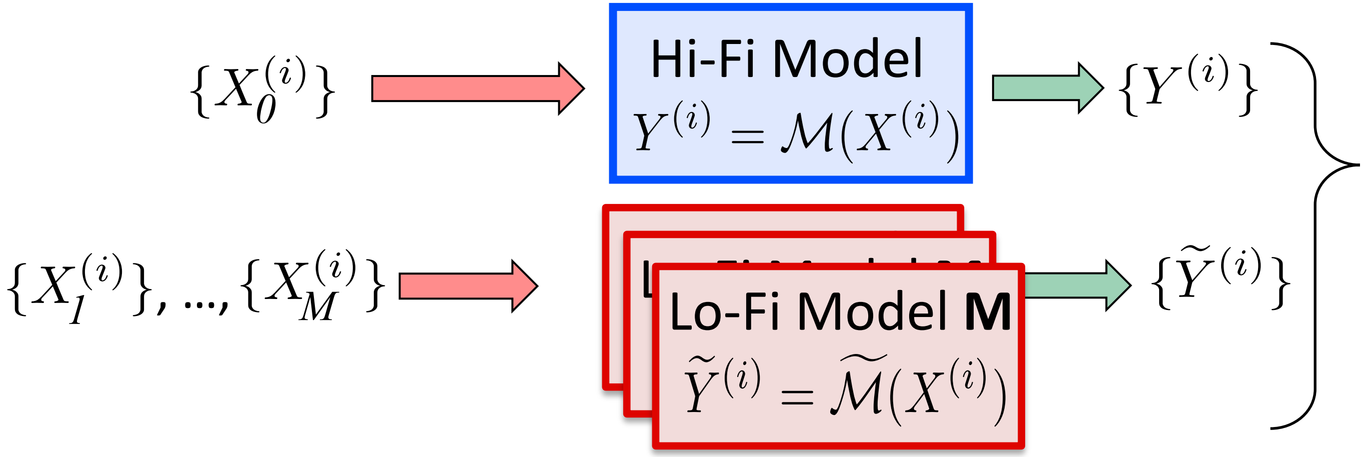
* Giles, M B. Operations Research (2008)

Multi-Model MC (2/2)

1) Draw & allocate random inputs

2) Evaluate models

3) Form estimator [3]



$$E[Y] \approx \widehat{Y}_0(X_0) + \sum_{i=1}^{M} \alpha_i \left(\widehat{Y}_i(X_i^*) - \widehat{Y}_i(X_i) \right)$$

- Lo-fi models are useful if they are correlated to hi-fi model and faster to evaluate
- **Key challenge**: solve sample allocation optimization to find find $\boldsymbol{X_0}, \boldsymbol{X_i^*}, \boldsymbol{X_i}$ such that estimator variance (MSE) is minimal and cost is under budget
- Existing methods differ on optimization strategy:

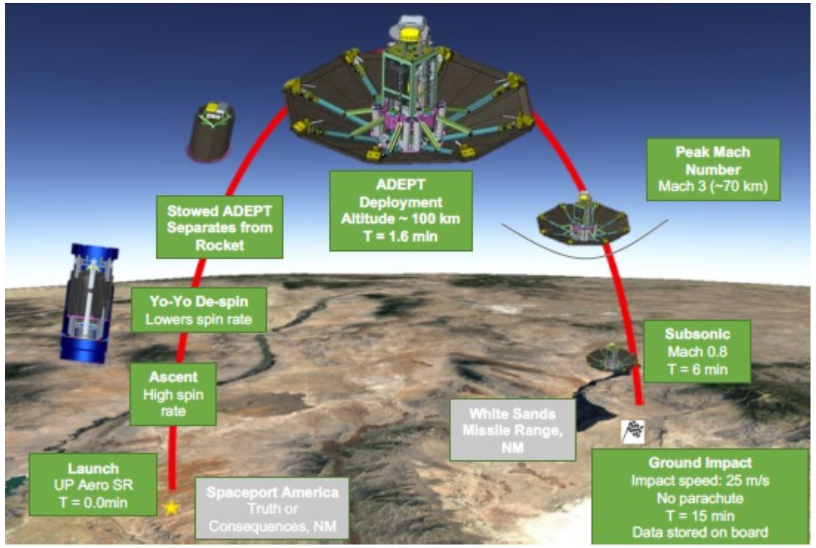
[1] Multilevel Monte Carlo (MLMC). Giles, 2008.
[2] Multifidelity Monte Carlo (MFMC). Peherstorfer, 2016.
[3] Approximate Control Variates (ACV). Gorodetsky, 2020.
[4] Parametrically-defined ACV. Bomarito, 2020.

Multi-Model Monte Carlo with Python (MXMCPy)
https://github.com/nasa/mxmcpy

Application: Trajectory Simulation

Analysis of the Adaptable Deployable Entry & Placement Technology (ADEPT) Sounding Rocket 1 (SR-1) test flight

<u>ADEPT SR-1 Test Flight Schematic</u>

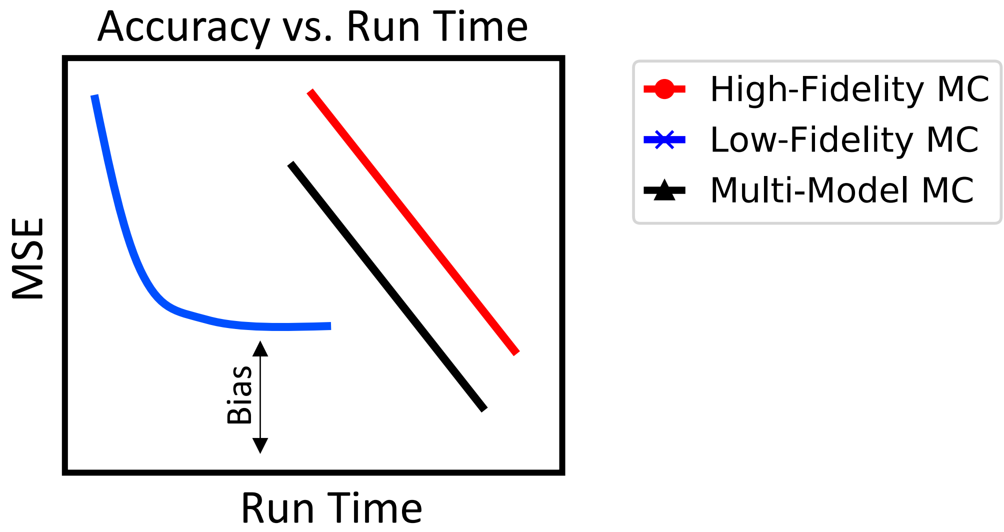


<u>Computational Models Considered</u>

Model	Description	Speedup
High Fidelity	POST2* w/ fine timestep	1X
Machine learning	POST2 surrogate model	~10000X
Coarse Time Step	100X larger timestep	~100X
Reduced Physics	Simplified atmosphere	~4X

* Program to Optimize Trajectories II (POST2). Langley Research Center.

- **Goal:** assess estimator accuracy for trajectory state quantities using multi-model MC versus both high & low fidelity MC

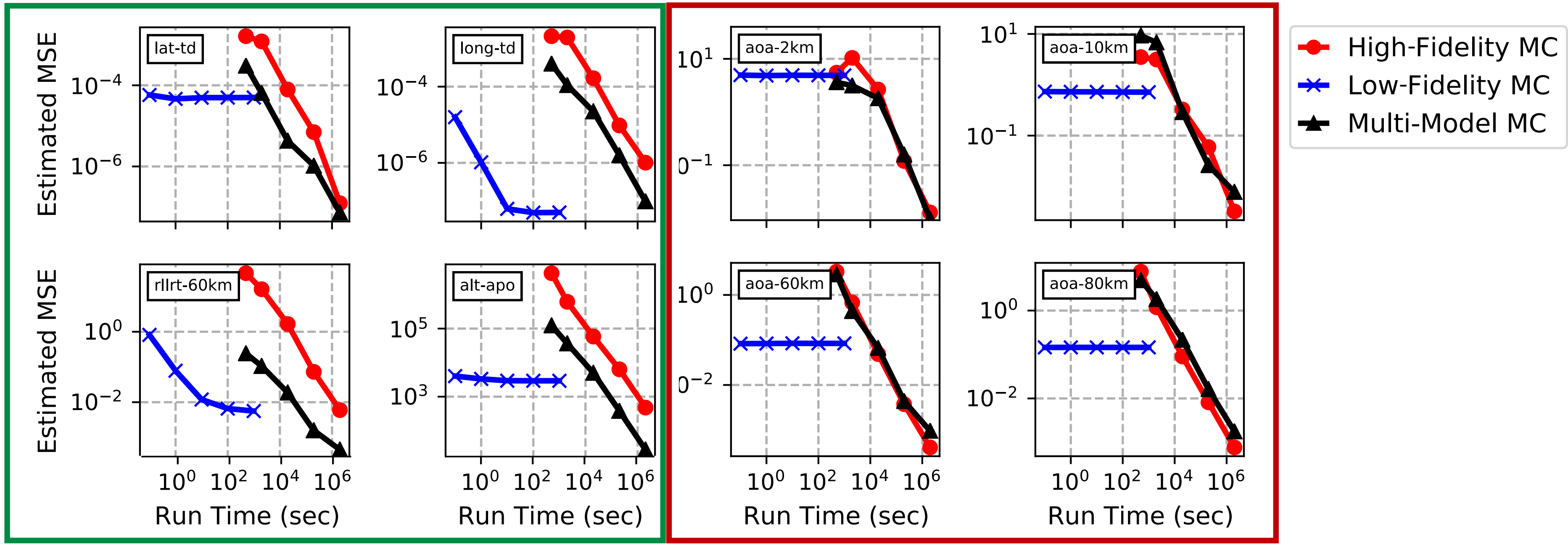


Mean Squared Error (MSE) = bias + variance

Estimator Accuracy vs. Run Time

- The MSE was calculated for high-, low-, and multi-model MC estimators versus a reference solution using high-fidelity MC with 100,000 samples

MSE versus Run Time for Several Trajectory QoIs



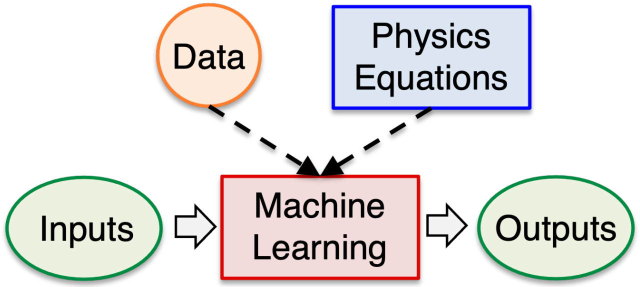
- ➢ Significant accuracy improvements for predictions of landing location, roll rate, and altitude
- ➢ Negligible improvements for predictions of angle of attack
  - **Cause: ineffective low-fidelity models**

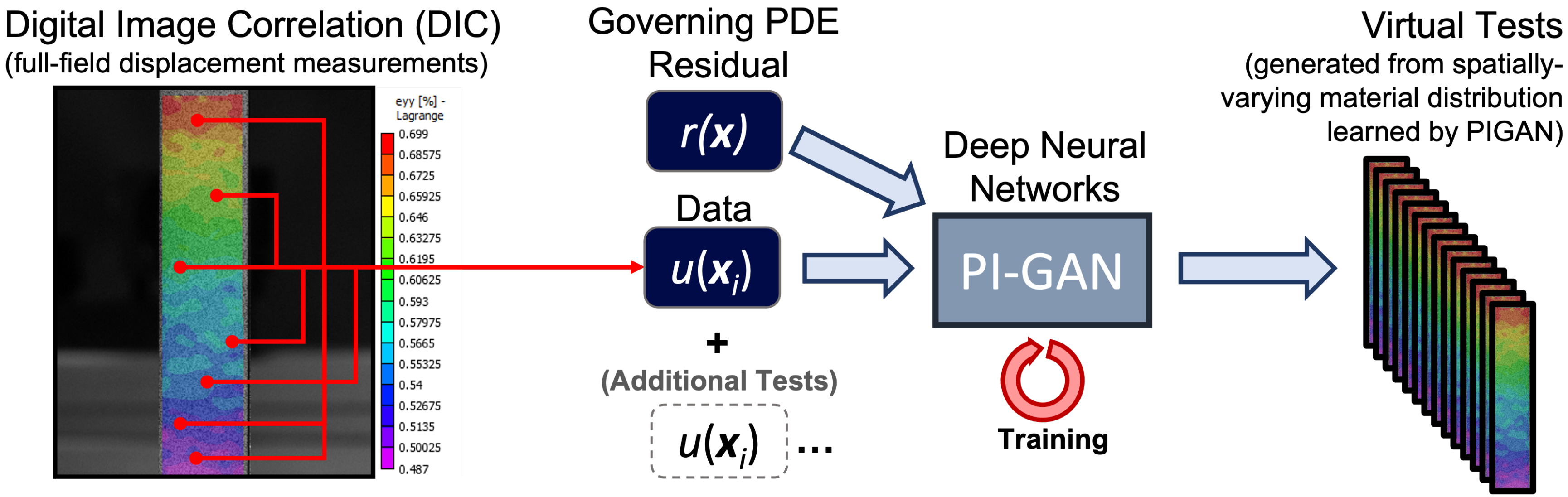
Multi-model MC: Summary & Future Work

- Combining hi-fi & lo-fi model predictions with multi-model MC has the potential to substantially improve accuracy for trajectory simulation
- Limitations & challenges:
  - Effectiveness is critically tied to development/selection of lo-fi models
  - Standard approaches are limited to statistical estimators (e.g., mean, variance)
- Future work:
  - Automating lo-fi model tuning & selection through collaboration with Sandia
  - Extending approach for higher-order statistics (e.g., confidence ellipses)
  - Embedding multi-model MC within robust trajectory optimization

Physics-Informed ML for Materials Identification

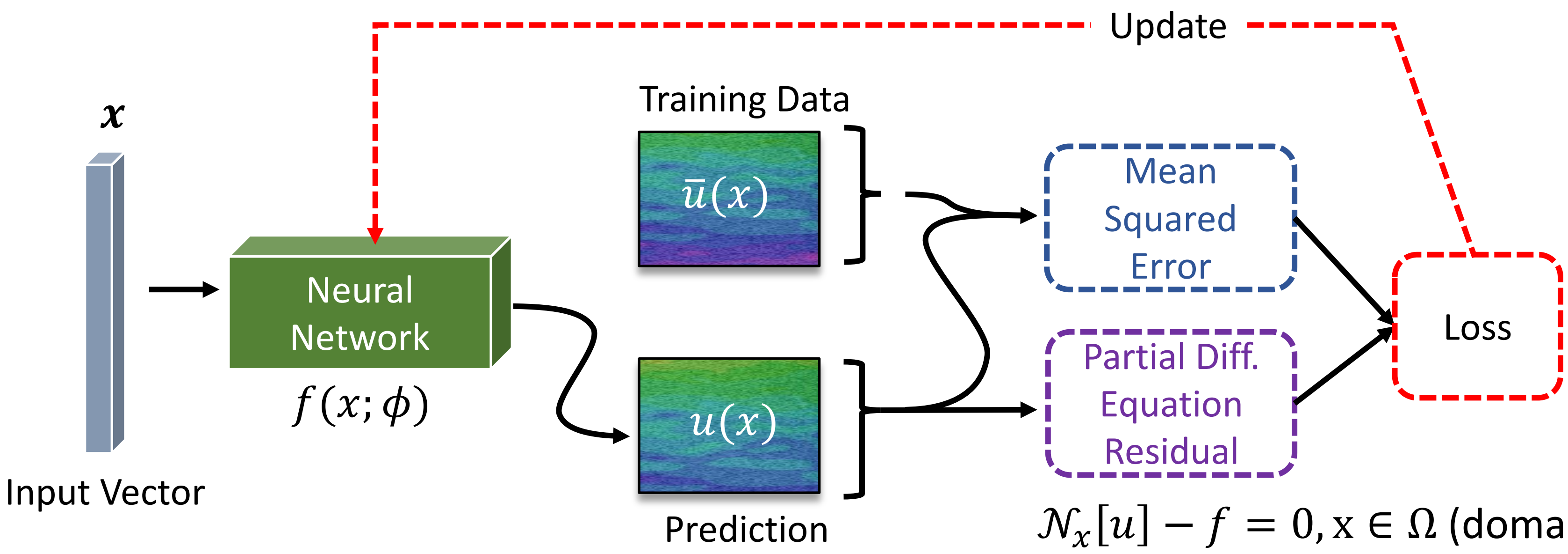
- **Motivation**: Materials characterization from limited tests
- **Challenge**: solving high-dimensional, probabilistic inverse problems
- **Approach**: Implement physics-informed generative adversarial network (PIGAN) capable of learning physically-admissible, random material property fields from data
- **Collaborators**: Julian Cuevas (U. Puerto Rico), Ted Lewitt (USC), Sean Lai (Portland State U.) – student interns at NASA





Physics-Informed Neural Networks (PINNs)

- Neural network trained to produce solution $u(x)$ to a partial differential equation
- Trained using two-part objective:
  1. Minimize mean squared error with respect to data
  2. Output physically admissible solutions throughout the problem domain

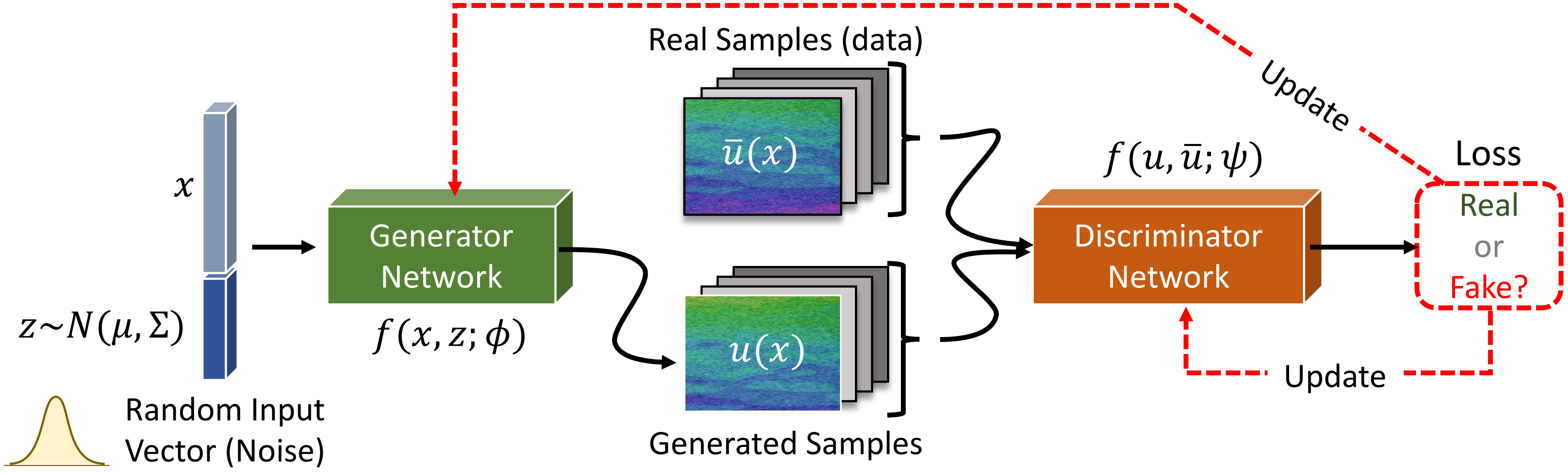


$\mathcal{N}_x[u] - f = 0, x \in \Omega$ (domain)
$B_x[u] - b = 0, x \in \Gamma$ (boundary)

Raissi et al. "Physics-informed neural networks: A deep learning framework for solving forward and inverse..." *J. Comp. Phys.* (2019)

Generative Adversarial Networks (GANs)

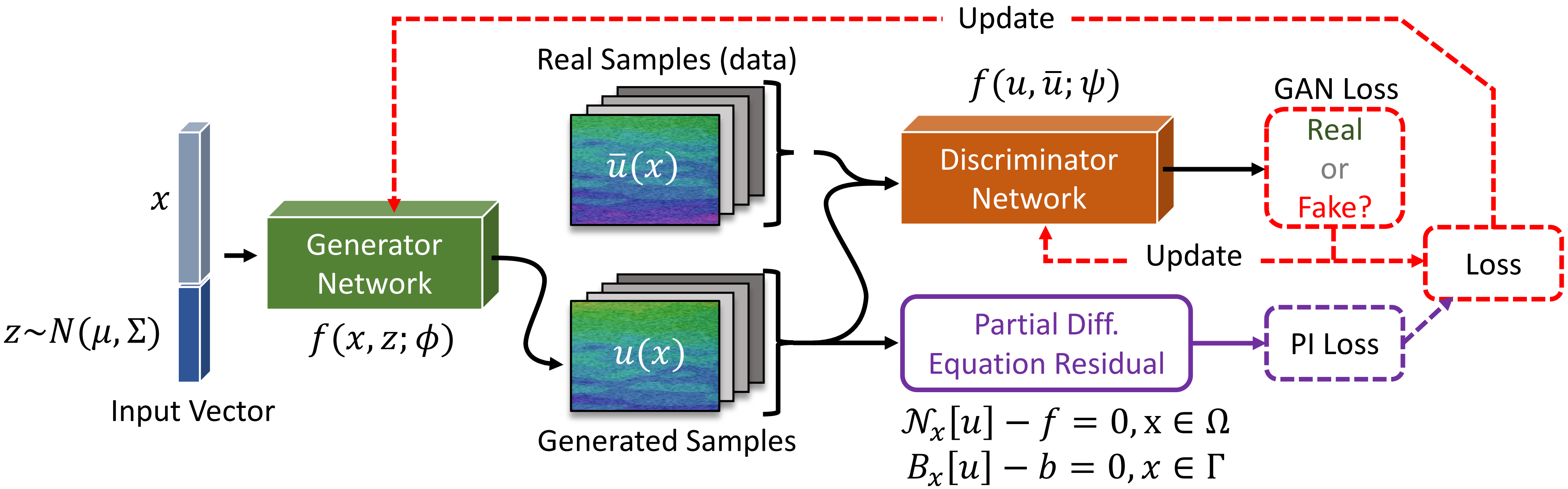
- Two neural networks compete in a zero-sum game
- Training objective: *discriminator network* tries to discern real from fake while *generator network* tries to fool the discriminator
- GANs are capable of learning/generating samples from complex probability distributions



Goodfellow, Ian J., et al. "Generative adversarial networks." arXiv preprint arXiv:1406.2661 (2014)

Physics-Informed GANS (PIGANs)

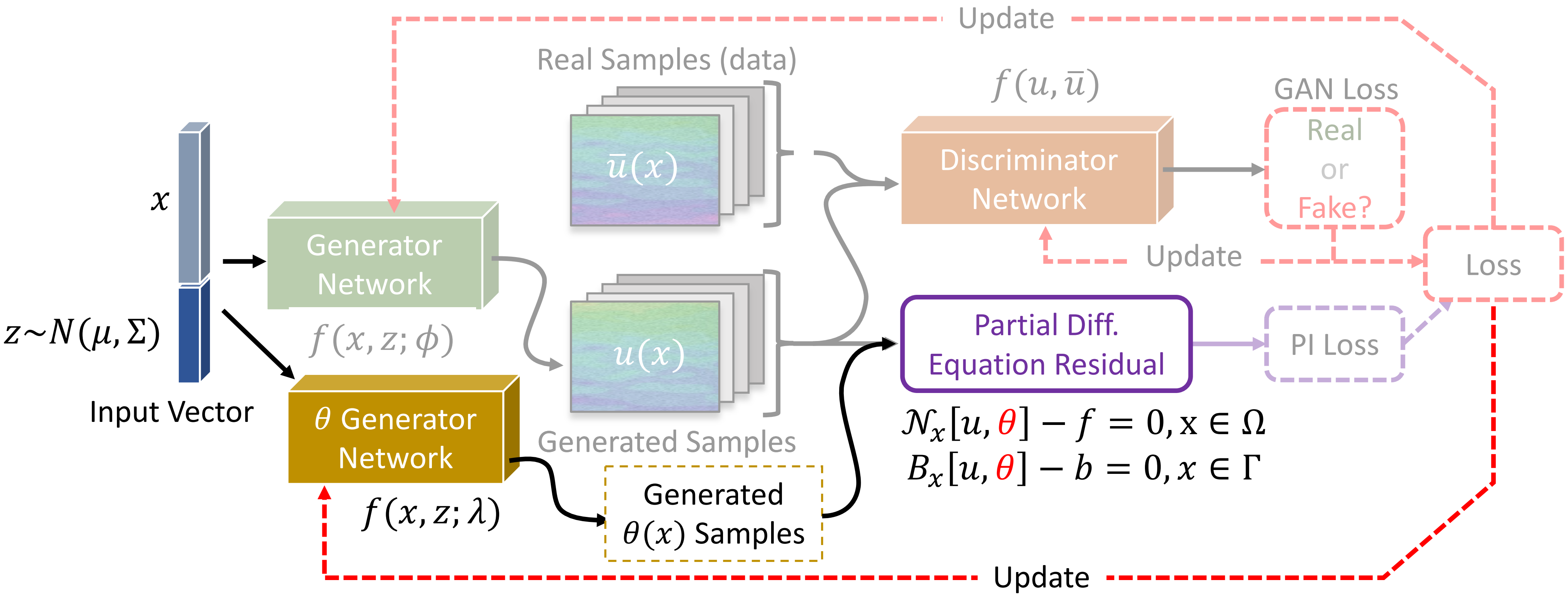
- Combination of PINNs and GANs
- Enables learning of physically admissible, *probabilistic* solution, $\boldsymbol{u}(x)$



Yang et al. "Physics-informed generative adversarial networks for stochastic differential equations." arXiv preprint arXiv:1811.02033 (2018)

Physics-Informed GANS (PIGANs)

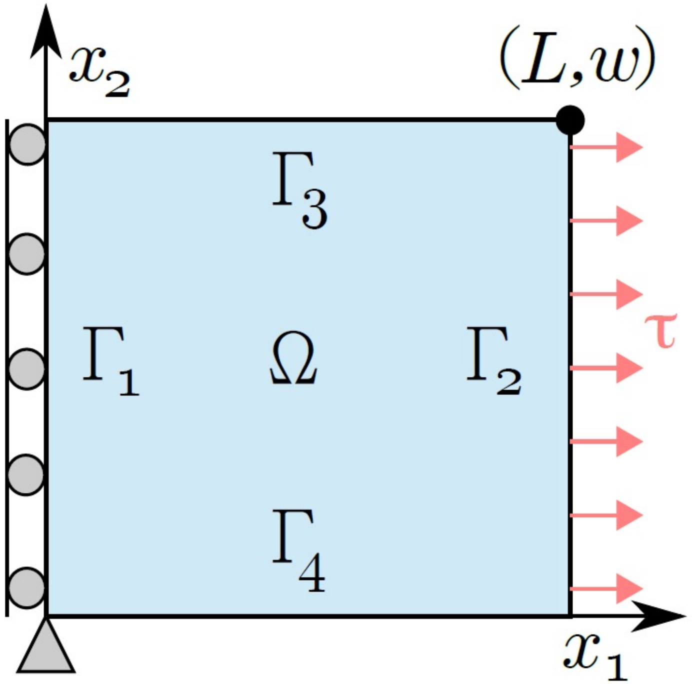
- Add generators for additional parameters, θ
- Enables solving complex, probabilistic inverse problems (no direct θ observations)



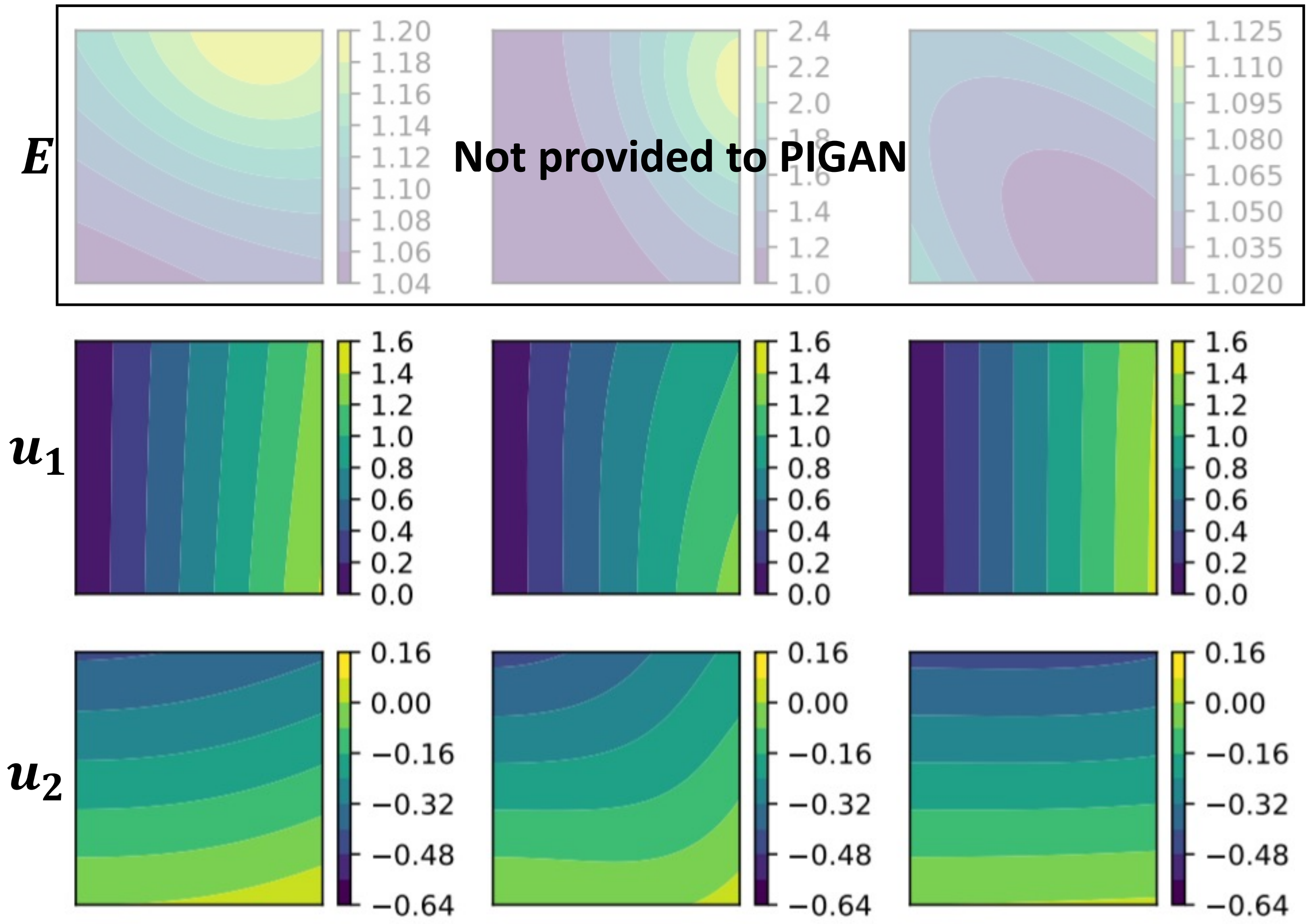
Yang et al. "Physics-informed generative adversarial networks for stochastic differential equations." arXiv preprint arXiv:1811.02033 (2018)

Example – 2D Plate in Tension with Synthetic Data

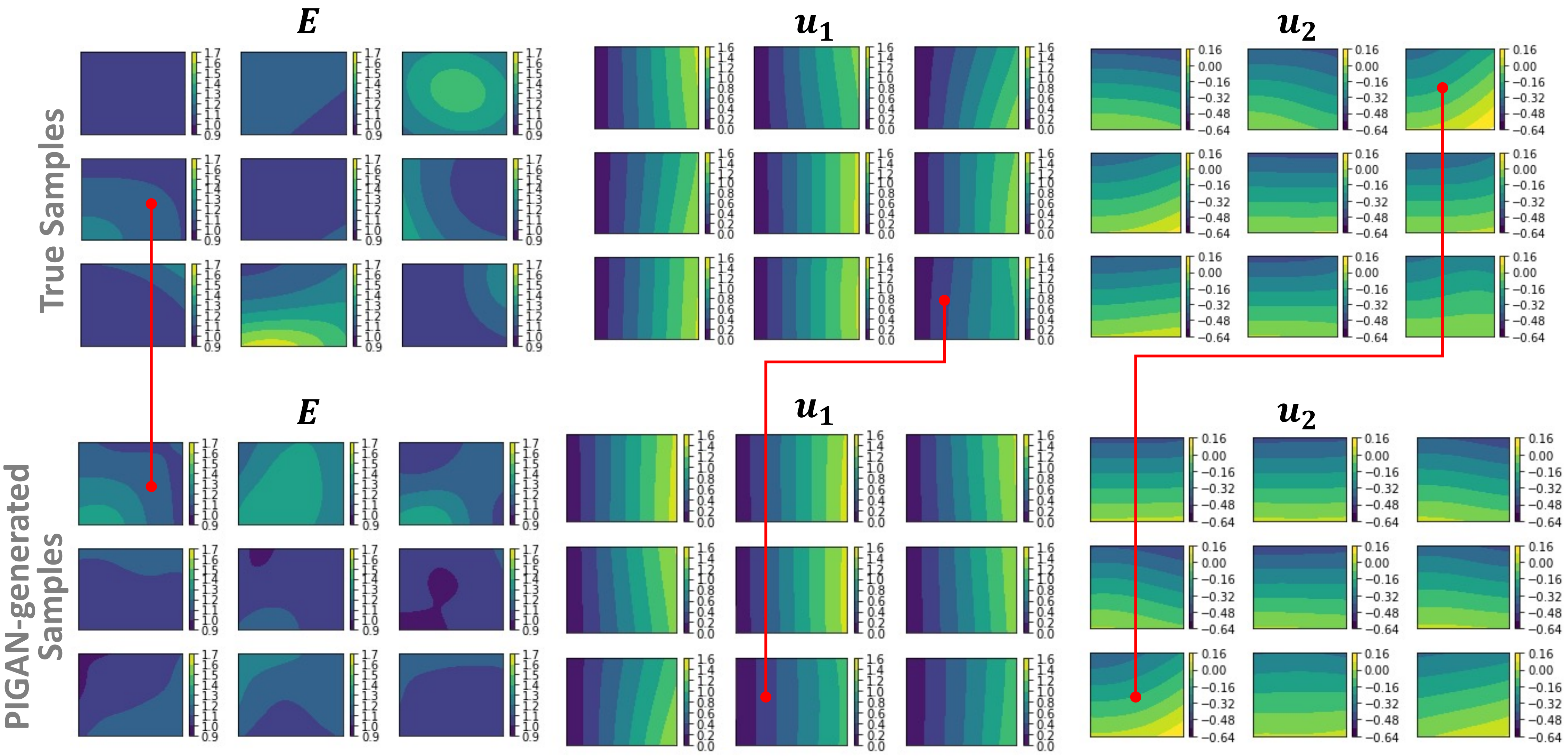
- 2D plate in tension with random, spatially-varying elastic modulus, $E(\boldsymbol{x})$
- Measurements of $\vec{u}(\boldsymbol{x})$ generated using Python finite element package FEniCS
- PIGAN learned both the $\vec{u}$ and E random fields



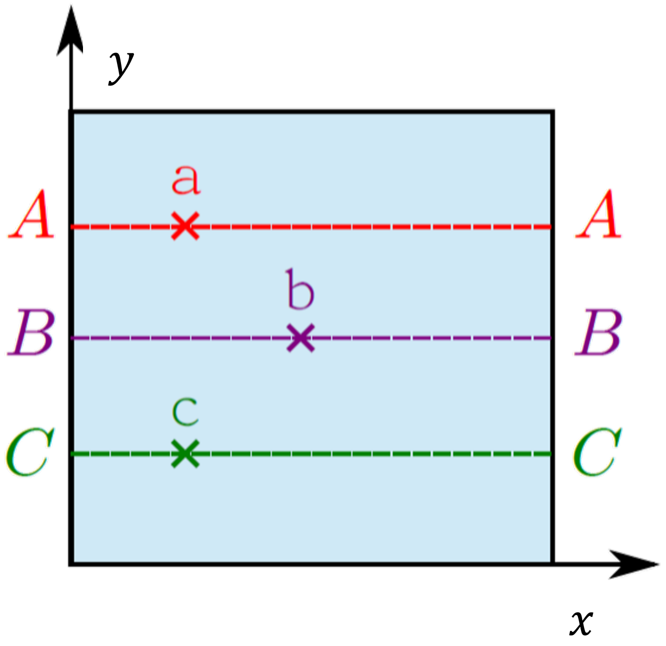
Randomly Generated Reference Fields (Data Snapshots)



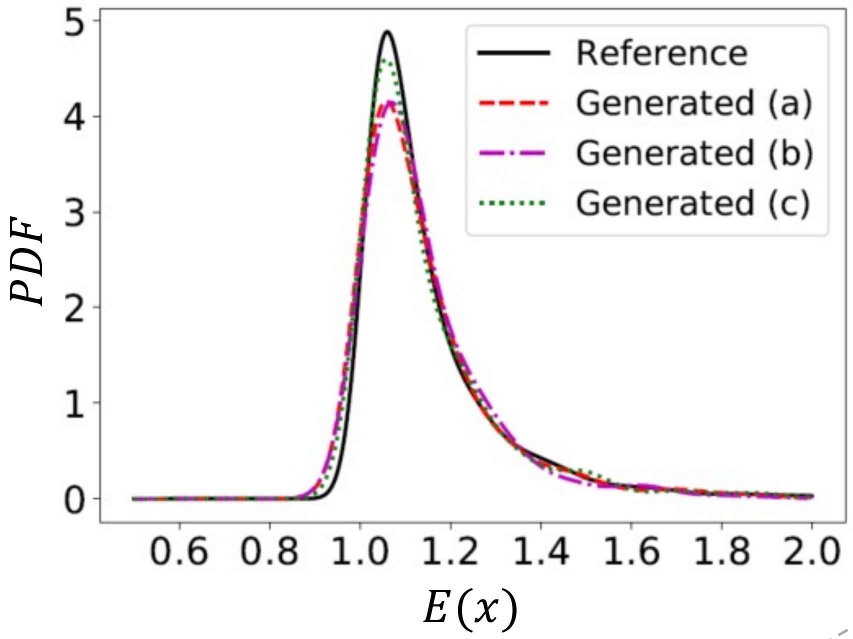
Example – Random Sample Comparison



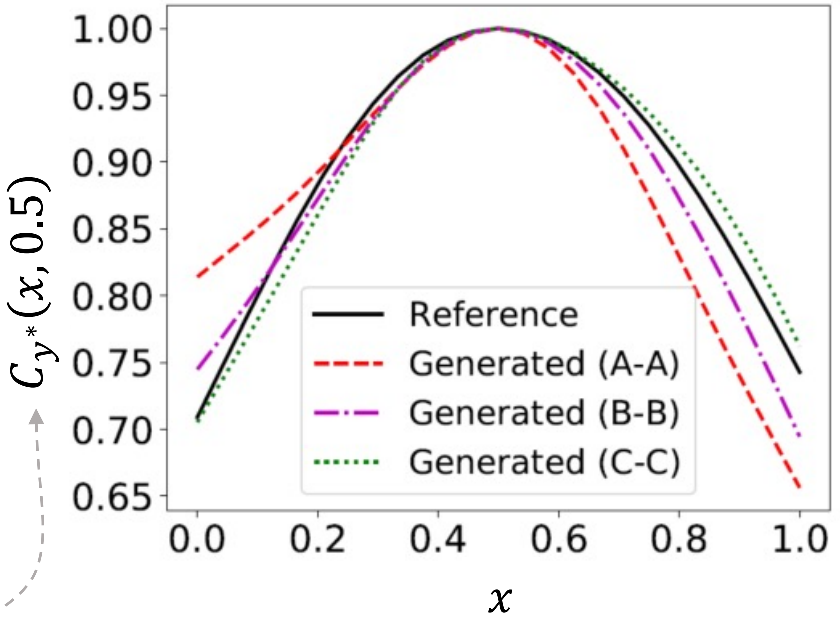
Example – Statistics Estimates



Probability Density Function



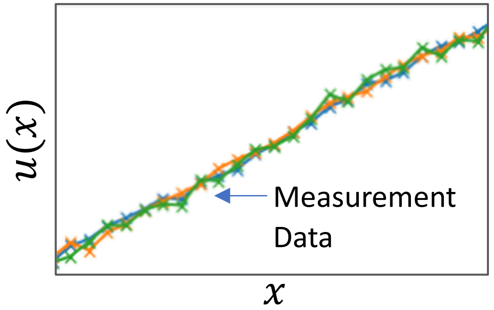
Correlation Function

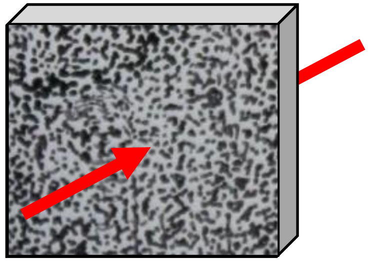


$C_{y^*}(x, 0.5) = \text{Correlation}[E(x, y^*), E(0.5, y^*)]$

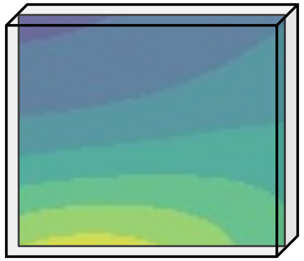
PIGANs: Summary & Future Work

- Demonstrated PIGANs as potential tool for solving high-dimensional, probabilistic inverse problems for material identification
- Limitations & challenges:
  - Case studies limited to simulated 2D problems with many measurements
  - Training can be computationally intensive and can exhibit instabilities
- Future work:
  - Training PIGANs in the presence of measurement noise
  - Solving for 3D displacement/property fields
  - Adaptively/automatically balancing training loss terms



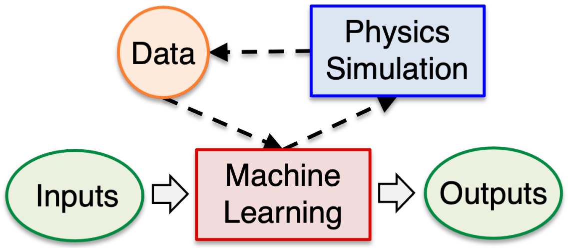


Front/back measurements



Internal solution (3D)

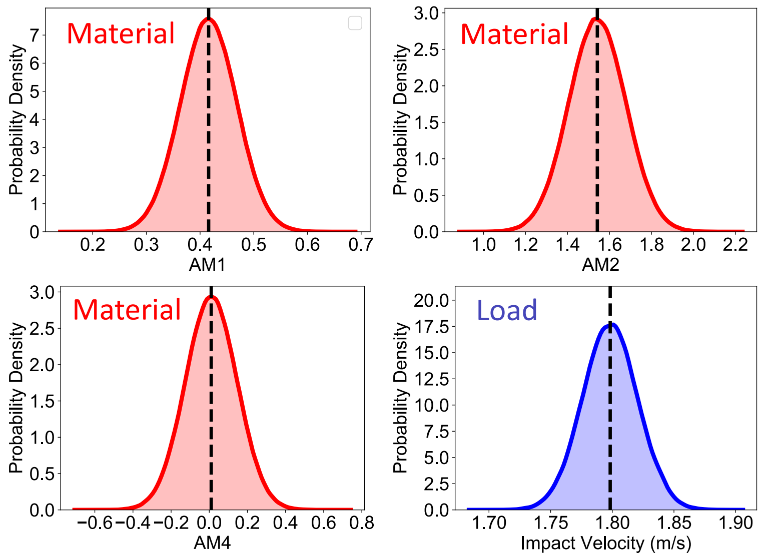
Active Learning for Reliability Analysis

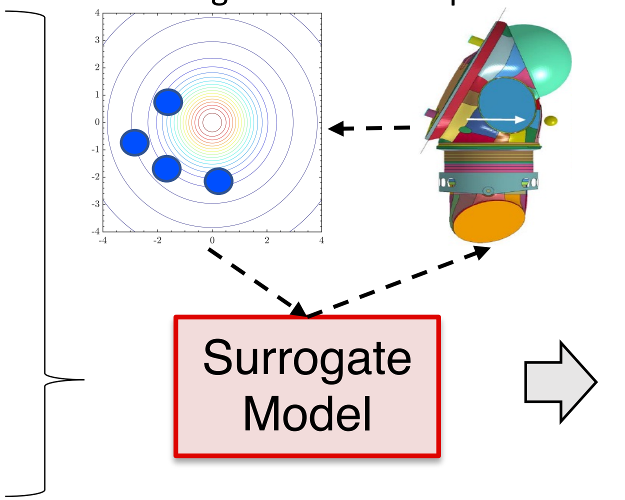


- **Motivation**: reliability analysis of the xEMU spacesuit
- **Challenge**: predicting small failure probabilities with expensive damage simulations
- **Approach**: use active learning to strategically select training data near failure regions for surrogate modeling
- **Collaborators**: Robert Gramacy, Austin Cole, Annie Sauer (Virginia Tech)

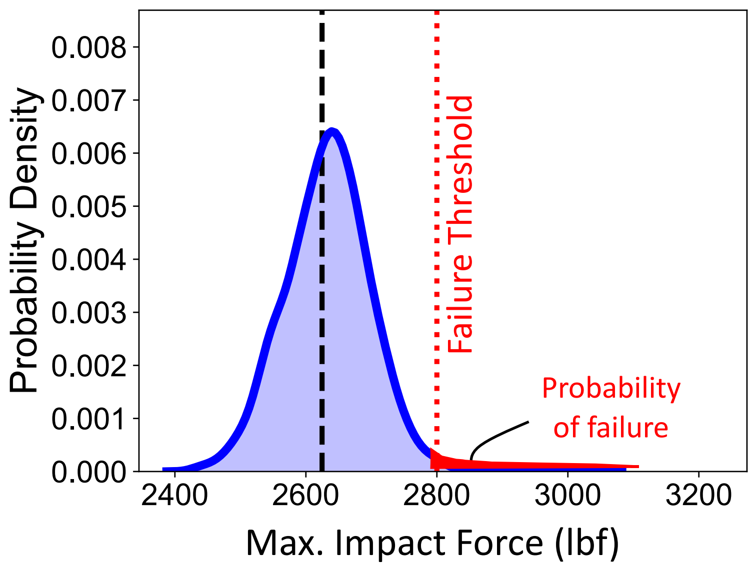
NASA Student Interns

Input Parameter Uncertainty





Max. Impact Force



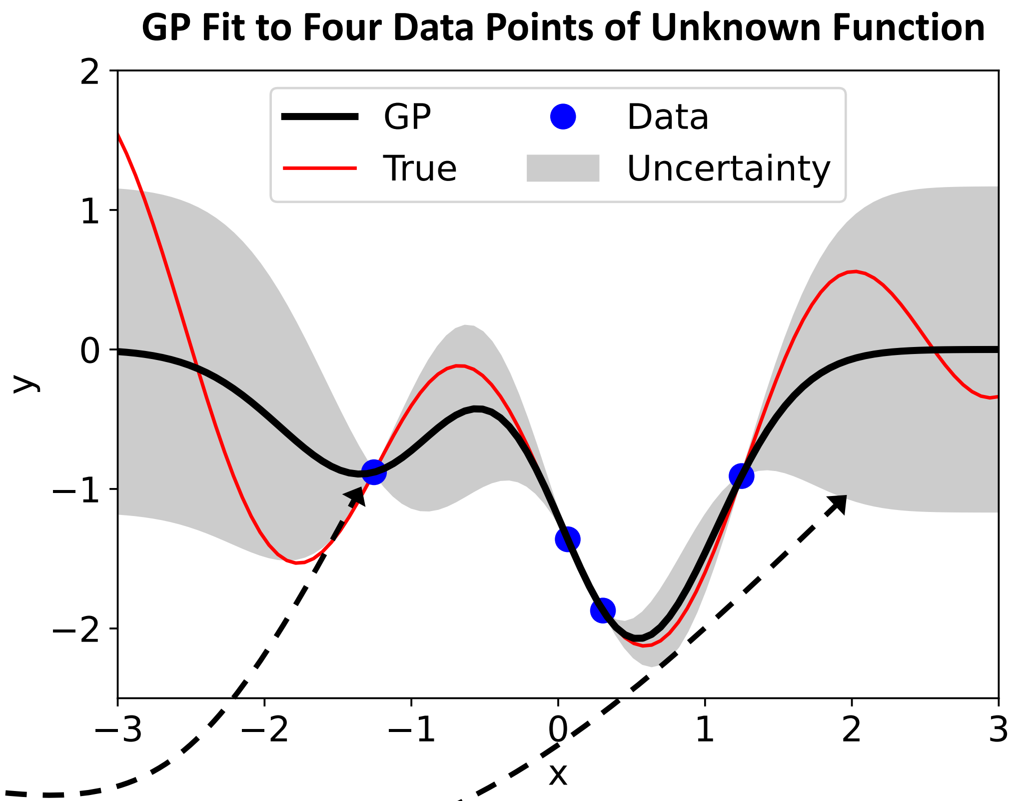
Gaussian Processes (GPs)

- Assume **N** training data, $(\mathbf{X}_N, Y_N)$
- A GP surrogate model, $\mathcal{S}_N$, provides a *probabilistic* prediction for a new input, **x'**, as the Gaussian distribution:

$$p(\mathbf{x}') = \mathcal{N}(\mu_N(\mathbf{x}'), \sigma_N^2(\mathbf{x}'))$$

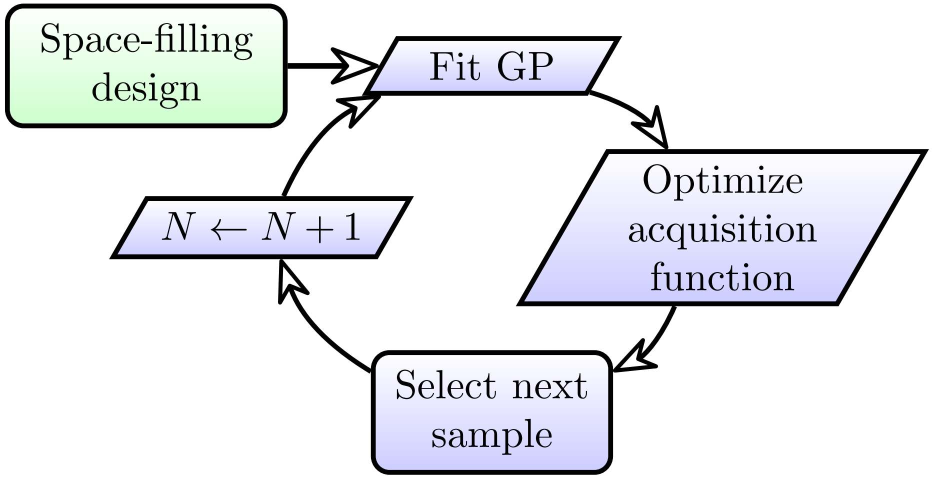
  - $\mu_N(\mathbf{x}')$: predictive mean
  - $\sigma_N^2(\mathbf{x}')$: predictive variance

- **Strengths:**
  - Interpolates observations of complex surfaces
  - Can compute pointwise confidence intervals



Active Learning

- *Active learning systems aim to make machine learning more economical, since they can participate in the acquisition of their own training data**



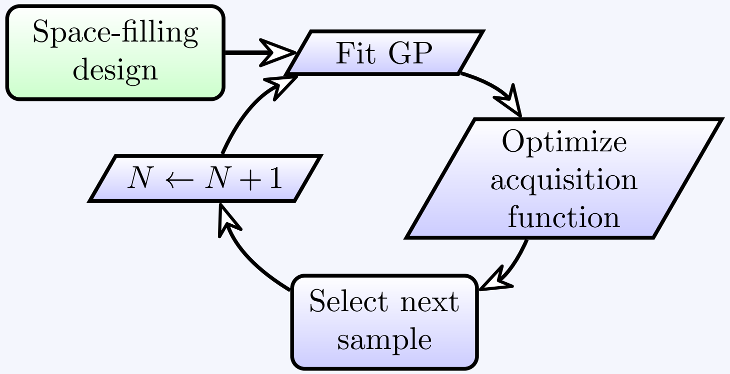
- Ex: integrated mean-squared error (IMSE) acquisition function minimizes future global variance:

$$\mathrm{IMSE}(\mathbf{x}_{N+1}) = \int_{\mathbf{x}\in\mathcal{X}} \sigma^2_{N+1}(\mathbf{x})d\mathbf{x}$$

$$\mathbf{x}_{N+1} = \mathrm{argmin}_{\mathbf{x}\in\mathcal{X}}\mathrm{IMSE}(\mathbf{x} \mid \mathcal{S}_N)$$

*B. Settles. From Theories to Queries: Active Learning in Practice. 2011

Active Learning - Demo

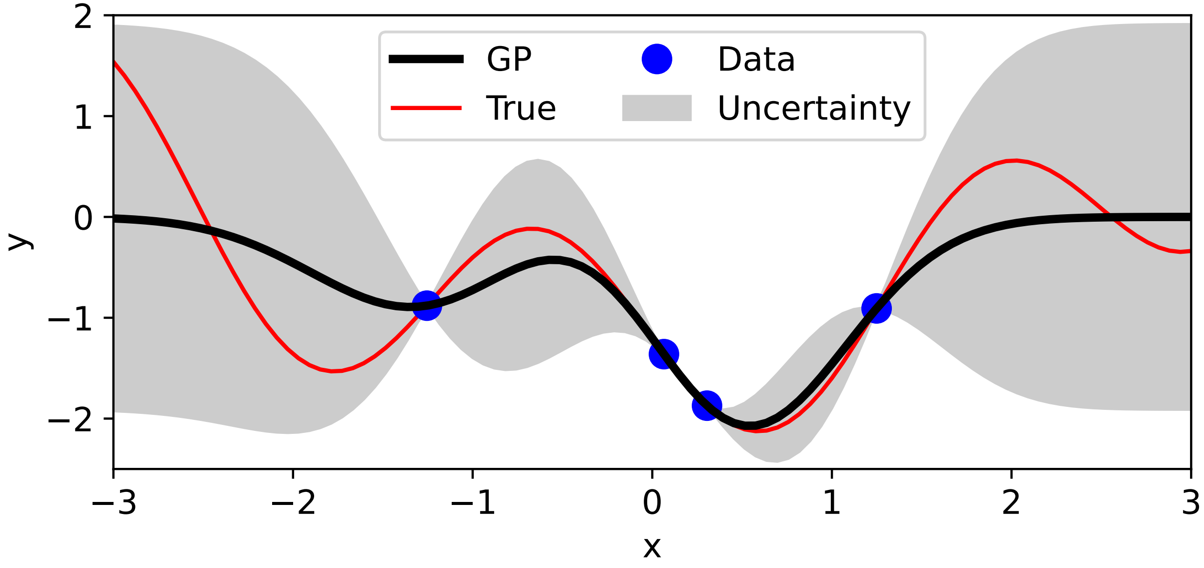


Acquisition function:

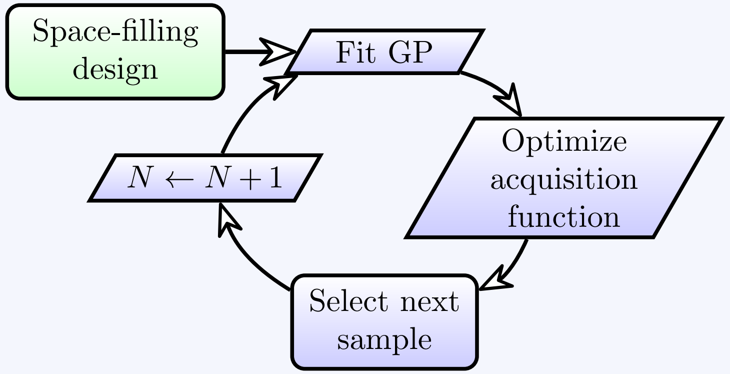
$$\mathrm{IMSE}(\mathbf{x}_{N+1}) = \int_{\mathbf{x}\in\mathcal{X}} \sigma^2_{N+1}(\mathbf{x})d\mathbf{x}$$

Select next sample:

$$\mathbf{x}_{N+1} = \mathrm{argmin}_{\mathbf{x}\in\mathcal{X}} \mathrm{IMSE}(\mathbf{x} \mid \mathcal{S}_N)$$



Active Learning - Demo

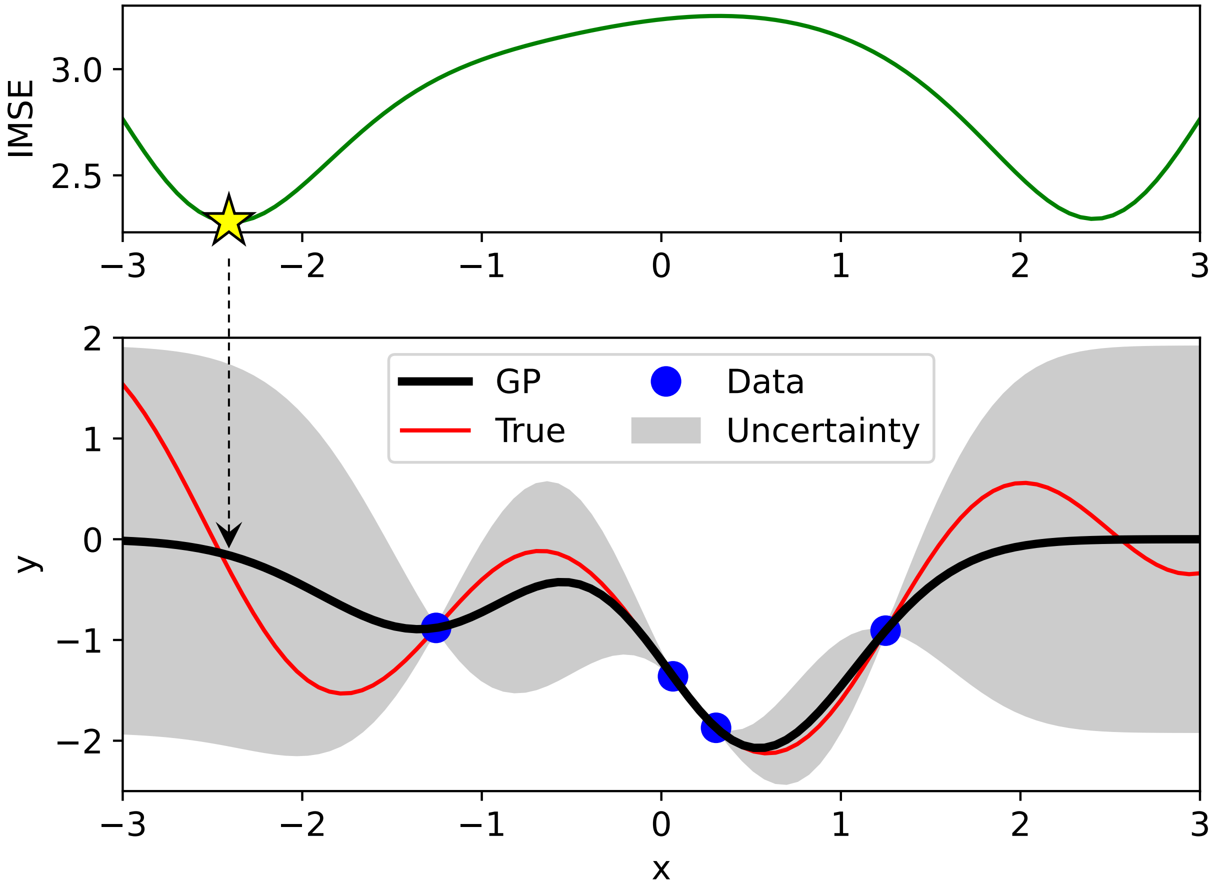


Acquisition function:

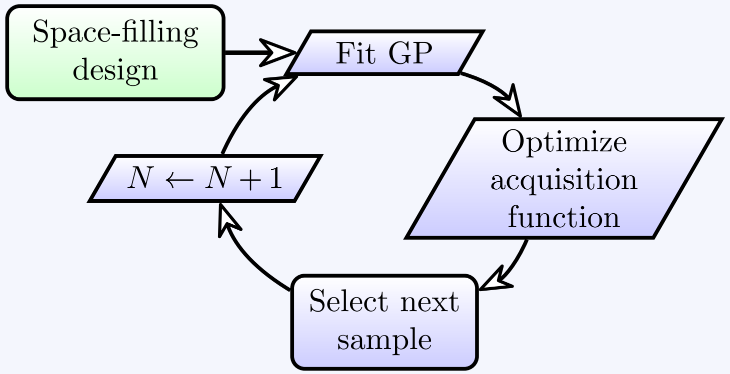
$$\mathrm{IMSE}(\mathbf{x}_{N+1}) = \int_{\mathbf{x}\in\mathcal{X}} \sigma^2_{N+1}(\mathbf{x}) d\mathbf{x}$$

Select next sample:

$$\mathbf{x}_{N+1} = \mathrm{argmin}_{\mathbf{x}\in\mathcal{X}} \mathrm{IMSE}(\mathbf{x} \mid \mathcal{S}_N)$$



Active Learning - Demo

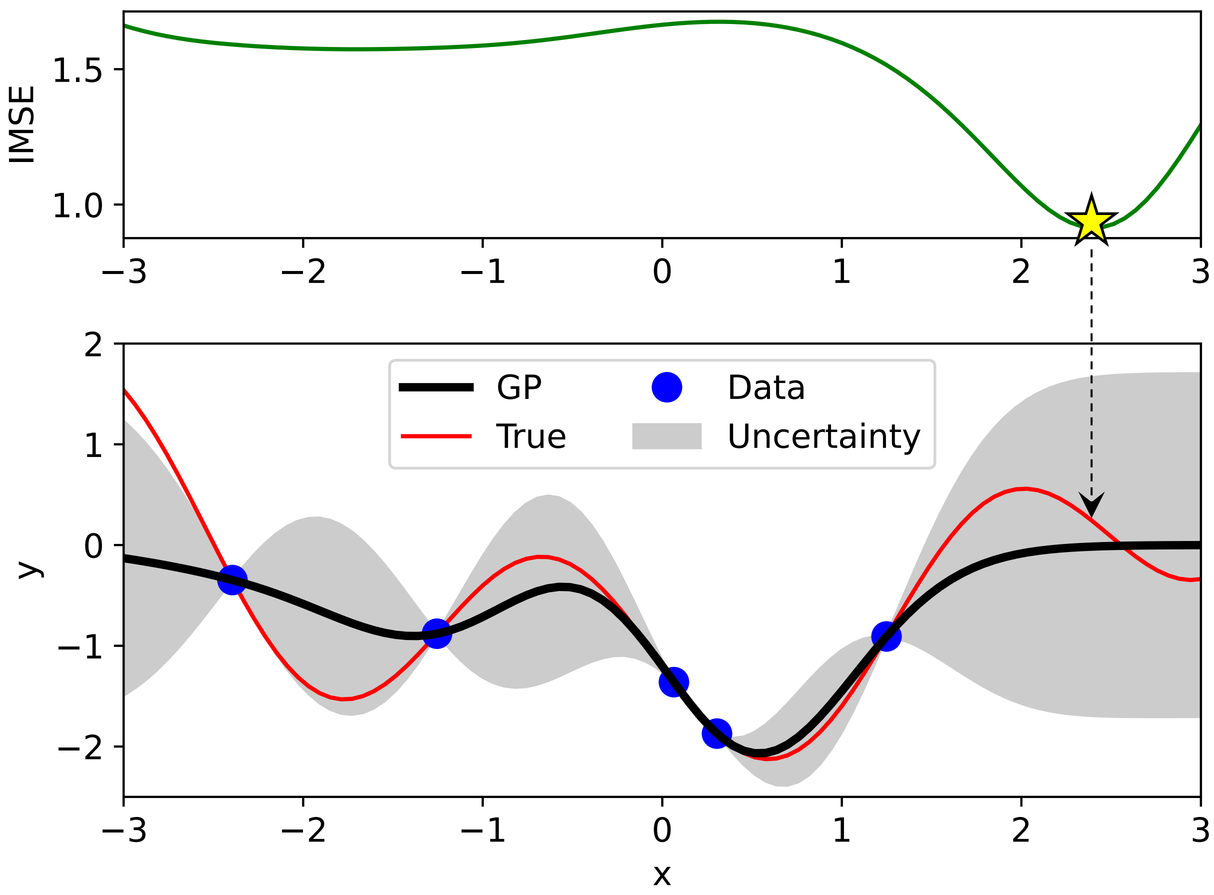


Acquisition function:

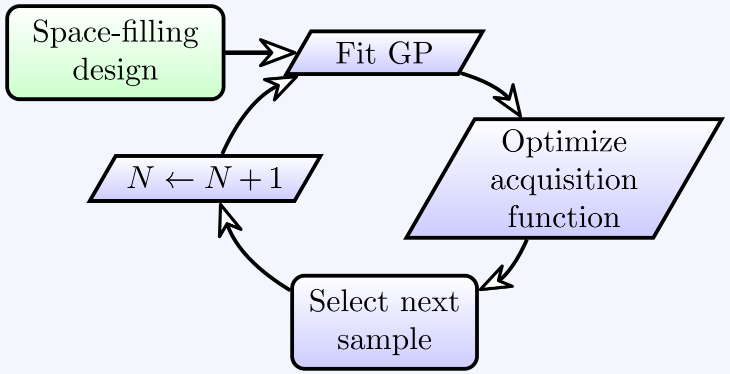
$$\mathrm{IMSE}(\mathbf{x}_{N+1}) = \int_{\mathbf{x}\in\mathcal{X}} \sigma^2_{N+1}(\mathbf{x})d\mathbf{x}$$

Select next sample:

$$\mathbf{x}_{N+1} = \mathrm{argmin}_{\mathbf{x}\in\mathcal{X}}\mathrm{IMSE}(\mathbf{x} \mid \mathcal{S}_N)$$



Active Learning - Demo



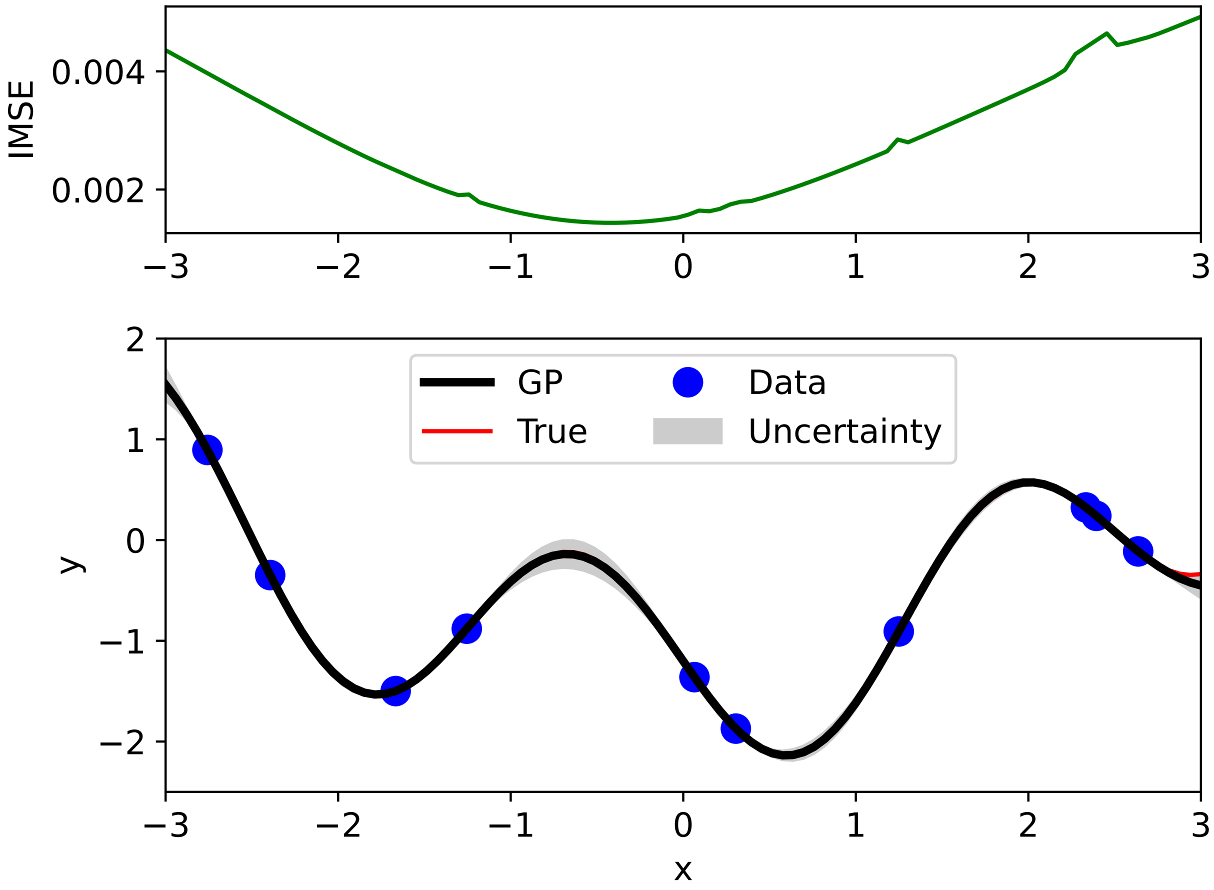
Acquisition function:

$$\mathrm{IMSE}(\mathbf{x}_{N+1}) = \int_{\mathbf{x}\in\mathcal{X}} \sigma^2_{N+1}(\mathbf{x})d\mathbf{x}$$

Select next sample:

$$\mathbf{x}_{N+1} = \mathrm{argmin}_{\mathbf{x}\in\mathcal{X}}\mathrm{IMSE}(\mathbf{x} \mid \mathcal{S}_N)$$

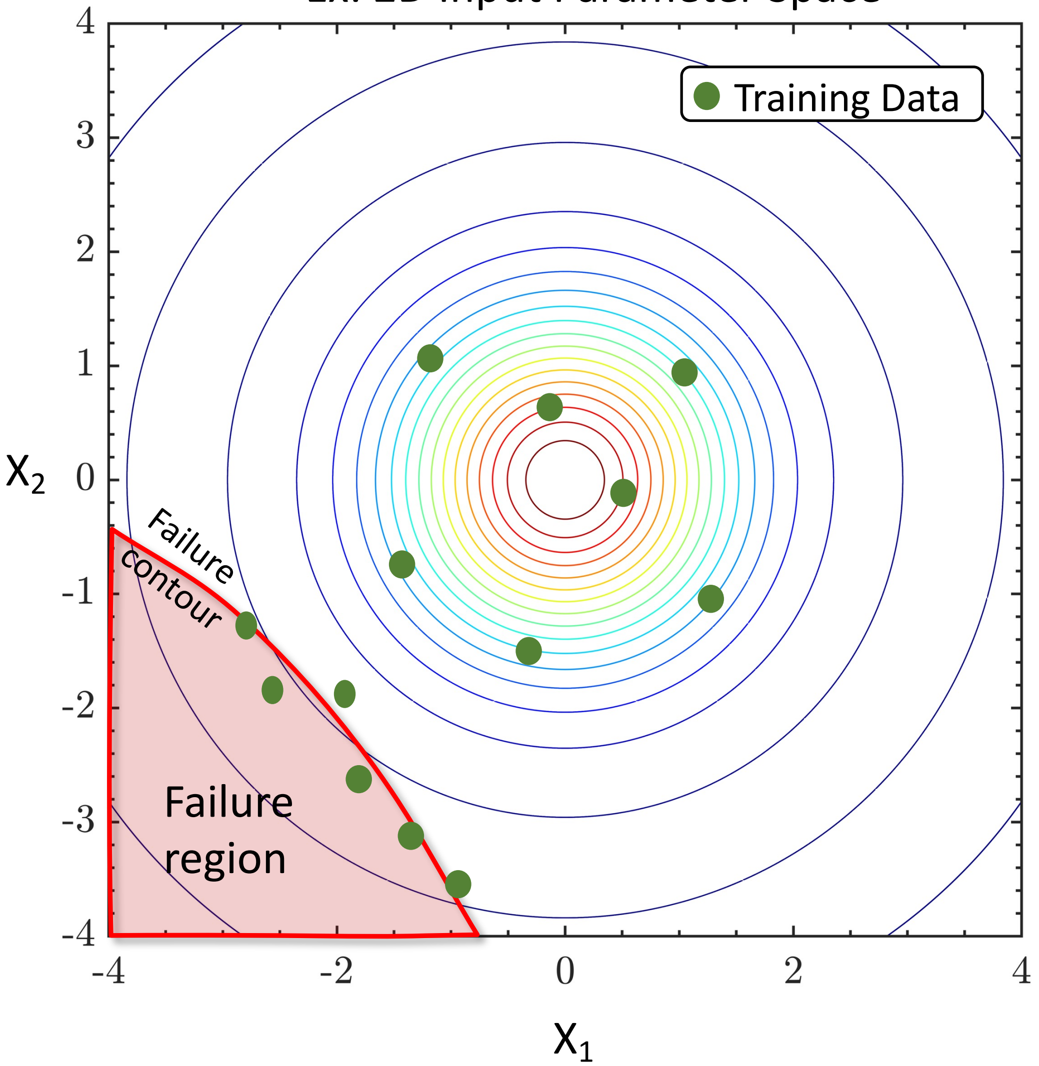
After six sequentially selected training data:



Active Learning for Identifying Failure Contours

- Introduced ***Entropy Contour Locator (ECL)**** acquisition function to concentrate training data near failure contours
  - Select locations of highest pass/fail uncertainty (maximum entropy)
- Resulting GP surrogate model will be tailored for accurate reliability analysis

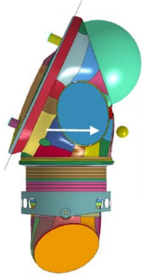
Ex: 2D Input Parameter Space



*D. A. Cole, R. B. Gramacy, J. E. Warner, G. F. Bomarito, P. E. Leser, W. P. Leser. *Entropy-Based Adaptive Design for Contour Finding and Estimating Reliability*. Journal of Quality Technology. 2021

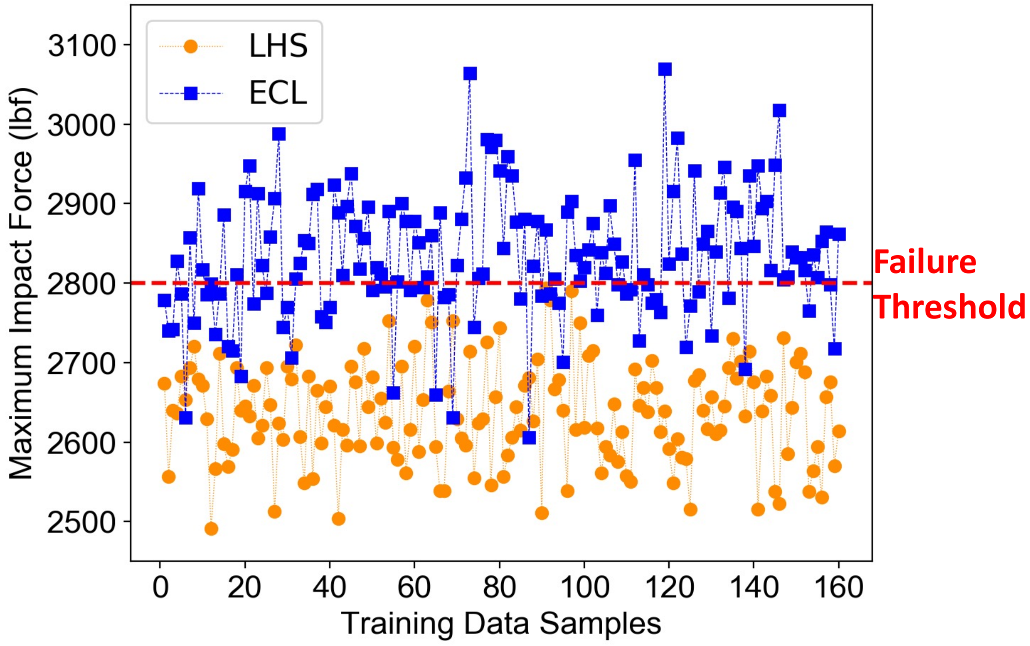
Spacesuit Impact Reliability

- **Goal**: estimate the probability of impact failure (contact force > 2800lbf) in the spacesuit under material/impact velocity uncertainty
- Compare surrogate-based solutions from:
  - **LHS GP** – trained from random **L**atin **H**yper**c**ube samples
  - **ECL GP** – trained from sequentially selected points targeting failure region

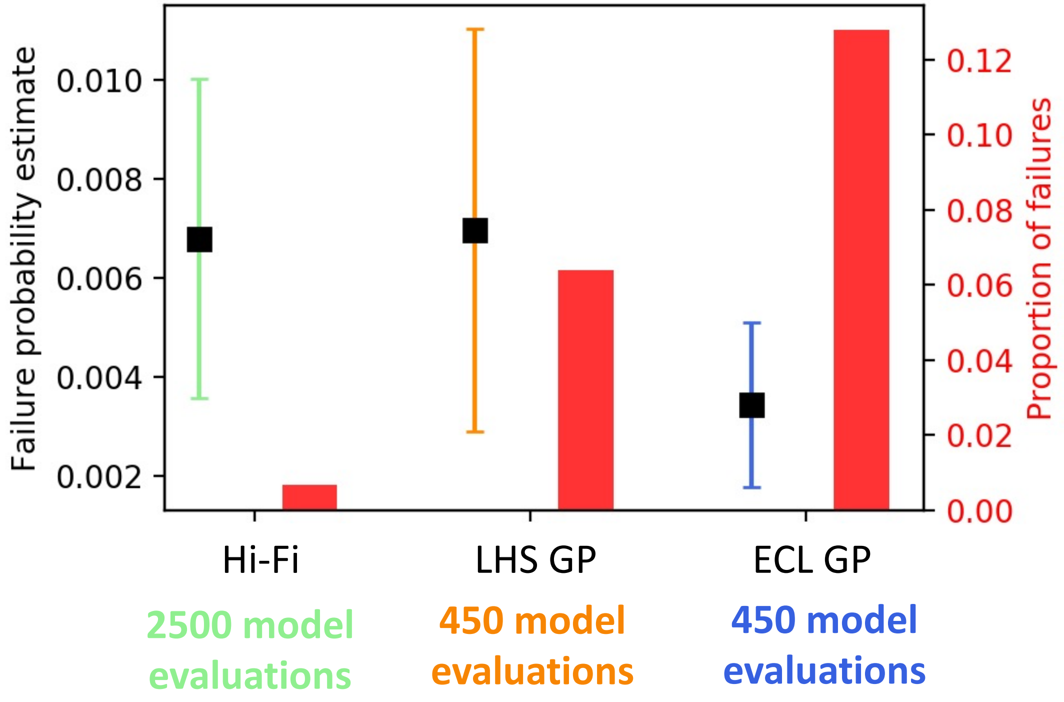


Model Run Time:
~18 hours

Training Data Selection: LHS vs. ECL

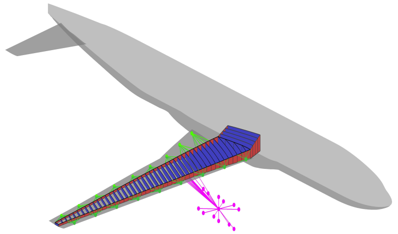


Probability of Failure Estimate Comparison



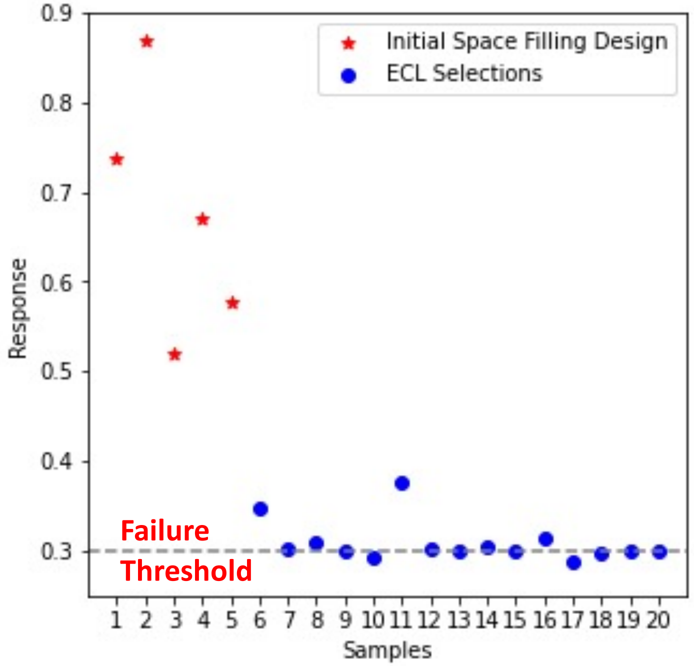
Aeroelastic Flutter Reliability

- **Goal**: estimate the probability of aeroelastic flutter under wing geometry & material uncertainties
- **Approach:** extend the ECL GP reliability method to use gradient observations produced by the transonic aero solver
  - Used only 20 total model evaluations to make predictions

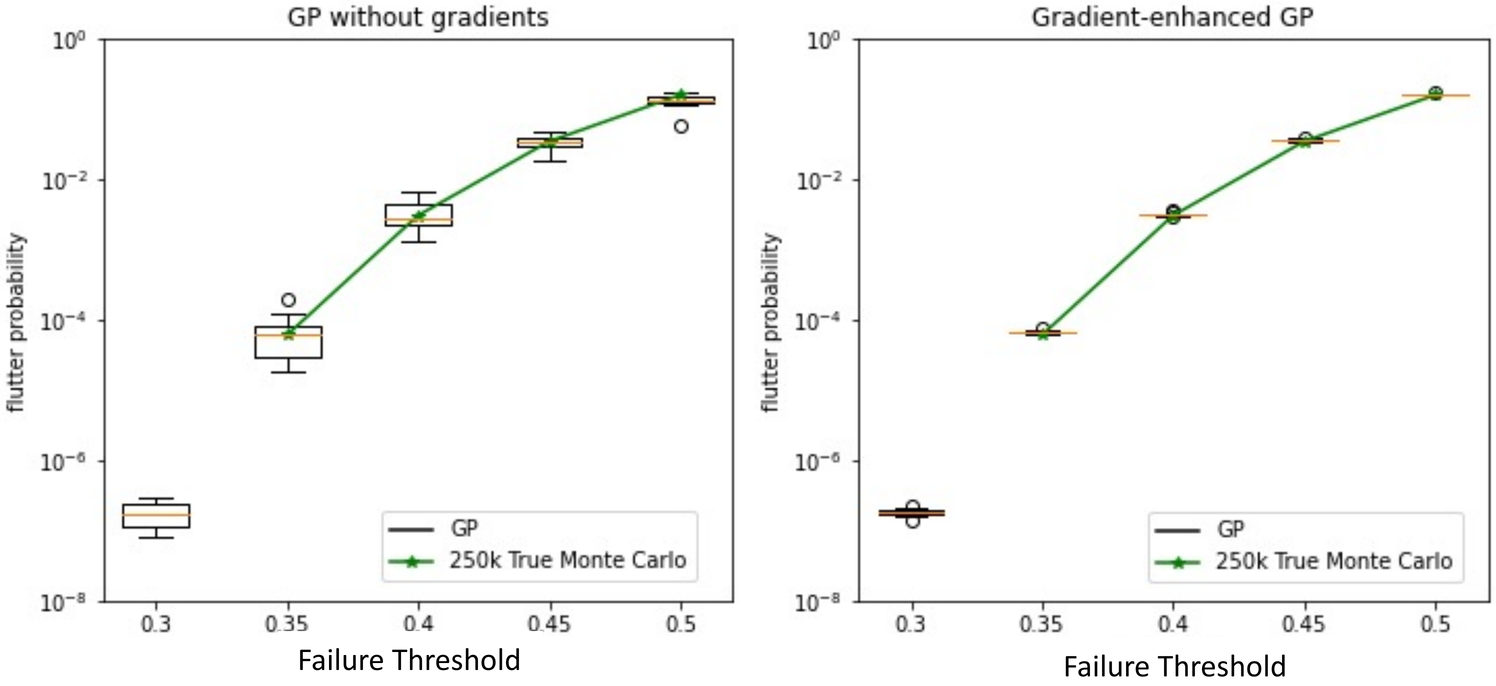


Model Run Time: ~3 minutes

Training Data Selection



Probability of Failure Predictions



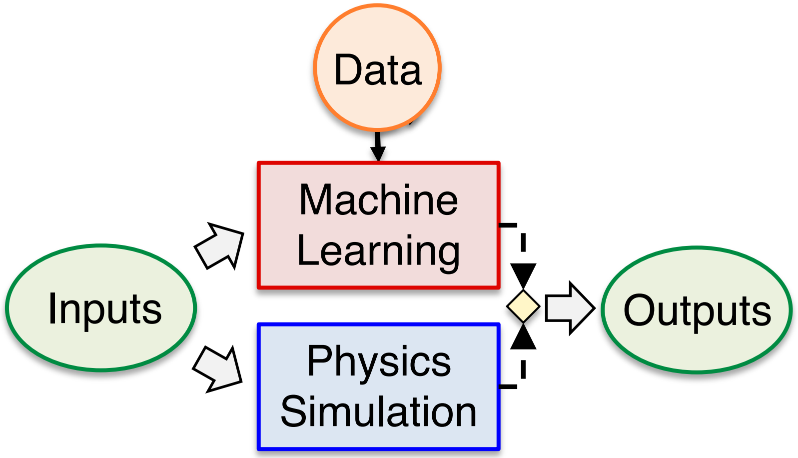
B. K. Stanford, K. E. Jacobson, A. Sauer, J. E. Warner. *Gradient-Enhanced Reliability Analysis of Transonic Aeroelastic Flutter*. 2022 AIAA Scitech Proceedings.

Summary

- Trusting black box models for UQ:
  - Use surrogate models responsibly:
    1. Always validate using a separate testing dataset
    2. Avoid extrapolation outside training data range
    3. Account for non-negligible surrogate errors in analysis
  - Combine physics-based & data-driven approaches:
    1. **Multi-model Monte Carlo**: fuse predictions from ML with physics-based models for unbiased *trajectory simulation* estimators
    2. **Physics-informed machine learning**: integrate physical laws into the training of ML models for *materials identification*
    3. **Active learning**: guide physics-based training data generation using ML models for *reliability analysis*

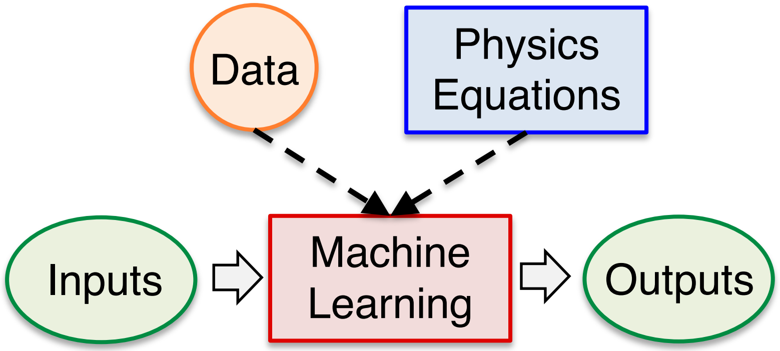
References

Multi-model Monte Carlo



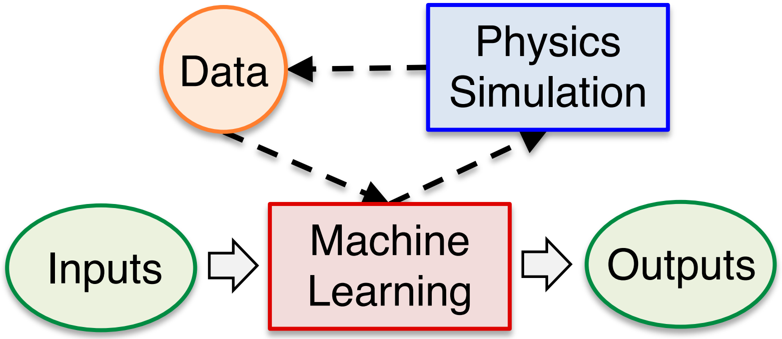
- J. E. Warner, G. F. Bomarito, L. Morrill, P. E. Leser, W. P. Leser, R. A. Williams, S. Dutta, S. Nieomoeller. *Multi-Model Monte Carlo Estimators for Trajectory Simulation*. 2021 AIAA SciTech Conference.
- G. F. Bomarito, P. E. Leser, J. E. Warner, W. P. Leser. *On the Optimization of Approximate Control Variates with Parametrically Defined Estimators*. Journal Paper. Journal of Computational Physics. February 2022.
- G. F. Bomarito, J. E. Warner, P. E. Leser, W. P. Leser, L. Morrill. Multi Model Monte Carlo with Python (MXMCPy). NASA TM 2020-220585. May 2020.

Physics-informed Deep Learning



- J. E. Warner, J. Cuevas, P. E. Leser, G. F. Bomarito, W. P. Leser. *Inverse Estimation of Elastic Modulus Using Physics-Informed Adversarial Networks*. NASA TM 2020-5001300. May 2020.

Active Learning



- D. A. Cole, R. B. Gramacy, J. E. Warner, G. F. Bomarito, P. E. Leser, W. P. Leser. *Entropy-Based Adaptive Design for Contour Finding and Estimating Reliability*. Journal of Quality Technology. In Review (Preprint available).
- B. K. Stanford, K. E. Jacobson, A. Sauer, J. E. Warner. *Gradient-Enhanced Reliability Analysis of Transonic Aeroelastic Flutter*. 2022 AIAA Scitech Proceedings.

Email: james.e.warner@nasa.gov